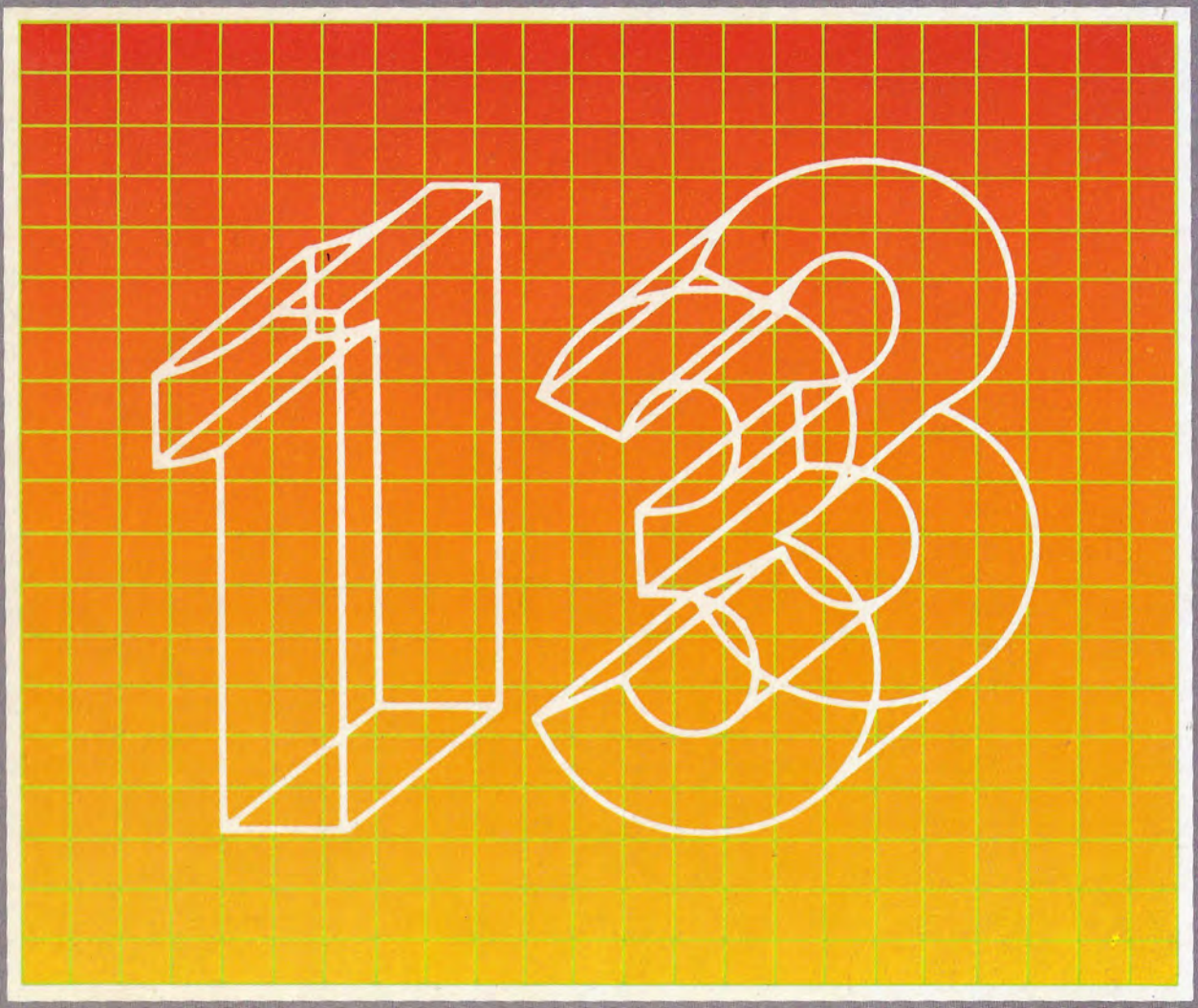
BIBLIOTECA PRACTICA

TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD, IBM, SPECTRUM, COMMODORE Y MSX



¿QUIERE APRENDER INFORMATICA?

PRACTIQUE CON SU ORDENADOR

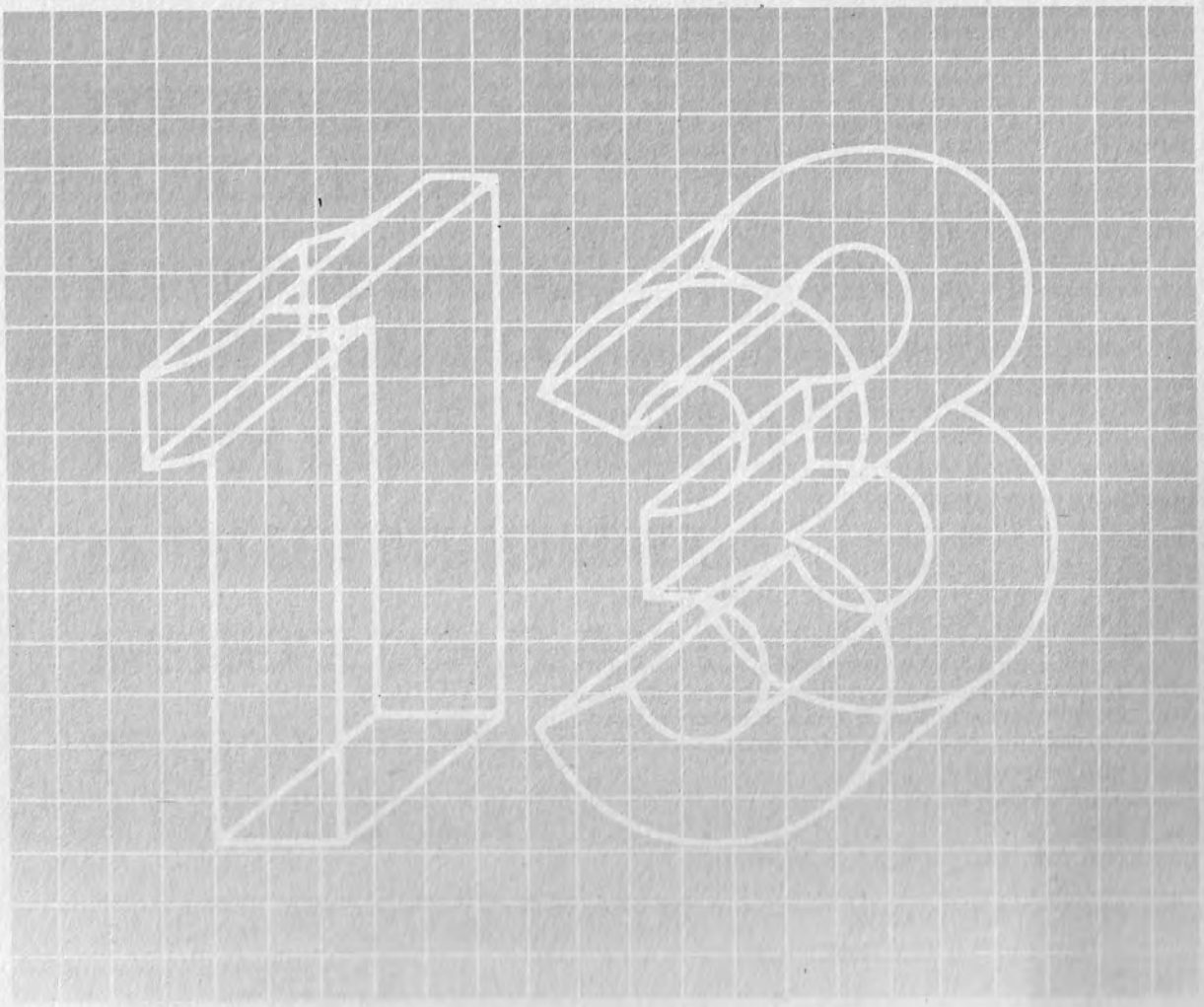
EDICIONES SIGLO CULTURAL

BIBLIOTECA PRACTICA

TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD, IBM, SPECTRUM, COMMODORE Y MSX



EDICIONES SIGLO CULTURAL

Una publicación de

EDICIONES SIGLO CULTURAL, S. A.

Director-editor:
RICARDO ESPAÑOL CRESPO.

Gerente:
ANTONIO G. CUERPO.

Directora de producción:
MARIA LUISA SUAREZ PEREZ.

Director de la colección:
JOSE ARTECHE, Ingeniero de Telecomunicación

Diseño:
BRAVO-LOFISH.

Maquetación:
D. S. M.

Dibujos:
JOSE OCHOA

Tomo 13. «Experiencias y prácticas en logo», Carlos Albert, Técnico de Informática; Adoración Llena Jubero, Técnico de Informática. «Manejo de sprites y elementos gráficos», «Trucos y rutinas básicas», Francisco Morales. Técnico de Informática. «Aprende con el ordenador», AULA DE INFORMATICA APLICADA: Soledad Tamariz-Martel, Diplomada en Telecomunicación; Francisco Blanca, Diplomado en Telecomunicación; Alejandro Marcos, Licenciado en Química; Fernando Suero, Diplomado en Telecomunicación. «El Taller del Hardware», Carlos Rey, Ingeniero Industrial. «Pequeña historia de la Informática», «Temas monográficos de vanguardia», «Terminología», «Vocabulario de Informática», Blanca Arbizu, Técnico de Informática.

Ediciones Siglo Cultural, S. A.

Dirección, redacción y administración:
Pedro Teixeira, 8-2.°. Teléf. 810 52 13. 28020 Madrid.

Publicidad:
Gofar Publicidad, S. A. Benito de Castro, 12 bis. 28028 Madrid.

Distribución en España:
COEDIS, S. A. Valencia, 245. Teléf. 215 70 97. 08007 Barcelona.
Delegación en Madrid: Serrano, 165. Teléf. 411 11 48.

Distribución en Ecuador: Muñoz Hnos.

Distribución en Perú: DISELPESA.

Distribución en Chile: Alfa Ltda.

Importador exclusivo Cono Sur:
CADE, S.R.L. Pasaje Sud América, 1532. Teléf.: 21 24 64.
Buenos Aires - 1.290. Argentina.

Todos los derechos reservados. Este libro no puede ser, en parte o totalmente, reproducido, memorizado en sistemas de archivo, o transmitido en cualquier forma o medio, electrónico, mecánico, fotocopia o cualquier otro, sin la previa autorización del editor.

ISBN del tomo: 84-7688-081-2
ISBN de la obra: 84-7688-047-2

Fotocomposición:
ARTECOMP, S. A. Albarracín, 50. 28037 Madrid.

Imprime:
MATEU CROMO. PINTO (Madrid).

© Ediciones Siglo Cultural, S. A., 1986

Depósito legal: M-43.185-1986
Printed in Spain - Impreso en España.

Suscripciones y números atrasados:
Ediciones Siglo Cultural, S. A.
Pedro Teixeira, 8-2.°. Teléf. 810 52 13. 28020 Madrid.

Abril, 1987.

P.V.P. Canarias: 355.—Ptas.

INDICE

EXPERIENCIA Y PRACTICAS EN LOGO 5

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS 15

TRUCOS Y RUTINAS BASICAS 21

EL TALLER DEL HARDWARE 27

APRENDER CON EL ORDENADOR 37

PEQUEÑA HISTORIA DE LA INFORMATICA 47

TEMAS MONOGRAFICOS DE VANGUARDIA 51

VOCABULARIO DE INFORMATICA 55

Procedimientos con más de dos entradas

NA vez que conocemos la forma de definir y trabajar con procedimientos de una y dos entradas, es fácil entender cómo se tiene que definir uno con más de dos entradas y cómo ejecutarlo.

A la hora de definir un procedimiento con más de dos entradas tenemos que dar un nombre a cada una de las variables que vamos a utilizar e indicarlos, por separado y precedidos de dos puntos, a continuación del nombre del procedimiento.

Por ejemplo, si deseamos definir un procedimiento en el que tres de los datos que utilizamos los queremos variar a nuestro gusto, lo haríamos de la siguiente manera:

```
? PARA Nombre Procedimiento :variable1 :variable2 :variable 3
```

Seguiremos añadiendo nombres de variables a medida que queramos o necesitemos utilizar más datos que actúen como variables.

Si, por ejemplo, queremos definir un procedimiento para dibujar un cuadrado en cualquier punto de la pantalla, de cualquir tamaño y en cualquier color, necesitamos utilizar cuatro variables: una para posicionar la Tortuga en un punto con respecto a la coordenada X, otra para posicionarla con respecto a la coordenada Y, una tercera para determinar lo que tiene que avanzar al dibujar cada lado y, por último, una cuarta para que dibuje el cuadrado con el color que deseemos:

```
? PARA CUA :PX :PY :LADO :COLOR
> SL
> PONX :PX
> PONY :PY
> PONCL :COLOR
> REPITE 4 [ AV :LADO GD 90 ]
> FIN
```

A la hora de ejecutar procedimientos con más de una entrada tenemos que dar valor a cada una de las variables que hayamos definido. Así, el ejemplo anterior podrá ejecutarse de la siguiente forma:

```
? CUA 40 -10 50 1
```

y obtenemos un cuadrado dibujado a partir del punto 40,-10, de 50 puntos de longitud y del color equivalente al código 1.

Utilizando varias entradas variables podemos ampiar los procedimientos y conseguir hacer con uno solo lo que sin variables haríamos con varios.

Veamos algún ejemplo de procedimientos con tres entradas:

— DIBUJAR Y COLOREAR BANDERAS

Definimos un procedimiento para dibujar la bandera.

```
? PARA BANDERA
> PONCL 1
> SL
> PONX -40
```

Un dato es un objeto Logo.

```
> BL
> PONY -75 REPITE 2 [ AV 150 GI 90
  AV 5 GI 90 ] AV 50
> REPITE 3 [ AV 25 BL REPITE 2 [ AV
  25 GD 90 AV 100 GD 90 ] SL ]
> FIN
```

Un procedimiento de tres entradas que llamamos COLOR, para colorear la bandera.

Vamos posicionando a la Tortuga en cada una de las franjas, ponemos el lápiz del color dado por las variables de entrada y rellenamos.

```
:A  color para franja superior
:B  color franja central
:B  color franja inferior

? PARA COLOR :A :B :C
> SL
> PONPOS [ 5 60 ]
> BL
> PONCL :A
> RELLENA
> SL
> PONY 40
> BL
> PONCL :B
> RELLENA
> SL
> PONY 20
> PONCL :C
> BL
> RELLENA
> FIN
```

DIBUJO inicializa la pantalla, dibuja la bandera de España y luego otra bandera, cuyos valores son asignados al azar.

```
? PARA DIBUJO
> BP
> OT
> BANDERA
> PONPALETA 0
> COLOR 2 3 2
> ESPERA 40
> BP
> BANDERA
> HAZ "A AZAR 4
> HAZ "B AZAR 4
> HAZ "C AZAR 4
> COLOR :A :B :C
> FIN
```

Ejecutamos el dibujo:

```
? DIBUJO
```

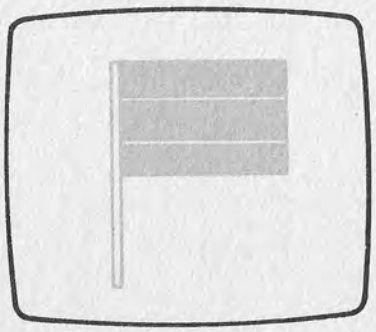


Fig. 1.

— DANDO EN EL BLANCO

Es un pequeño juego que consiste en rellenar de colores los cuadrados que aparecen en la pantalla, sin cambiar el color del marco y sin rellenar los espacios que hay entre las tiras de cuadrados.

Hemos intentado estructurar los procedimientos de tal forma que podáis, variando el número de cuadrados y su tamaño, aumentar o disminuir la dificultad del juego.

Para dibujar el marco de los cuadrados definimos el siguiente procedimiento:

```
? PARA MARCO
> SL PONPOS [ -125 -75 ]
> BL
> PONRUMBO 0
> REPITE 2 [ AV 165 GD 90 AV 240 GD
  90 ]
> FIN
```

Los cuadrados los dibujamos con este otro procedimiento de una entrada :Y que sirve para ir colocando en la coordenada Y las tiras de ocho cuadrados.

```
? PARA CUADRADOS :Y
> SL PONX -125 PONY :Y
> REPITE 8 [ BL REPITE 4 [ AV 30 GD
  90 ] HAZ "X COORX + 30 SL PONX
  :X ]
> FIN
```

Con DIBUJO, inicializamos la pantalla y dibujamos el tablero de juego llamando a MARCO y a CUADRADOS.

```
? PARA DIBUJOS :Y
> BP
> PM
```

Asignar un objeto a una variable es una operación sencilla.

```
> MARCO
> CUADRADOS 50
> CUADRADOS 10
> CUADRADOS -30
> CUADRADOS -70
> FIN
```

Y por último, APUNTA, que sirve para posicionar a la Tortuga en el punto que elijamos (X Y) y rellenar ese espacio con el color que hayamos indicado.

```
:X  posición coordenada X
:Y  posición coordenada Y
:C  código de color

? PARA APUNTA :X :Y :C
> SL PONX :X
> PONY :Y
> BL
> PONCL .C
> RELLENA
> FIN
```

Lo primero que debes hacer es dibujar el tablero de juego:

```
? DIBUJO
```

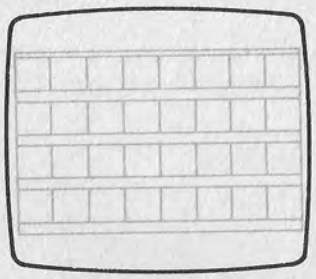


Fig. 2.

y ahora ya puedes apuntar:

```
? APUNTA 20 20 2
? APUNTA 20 60 1
? APUNTA 60 60 3
? APUNTA 100 60 2
```

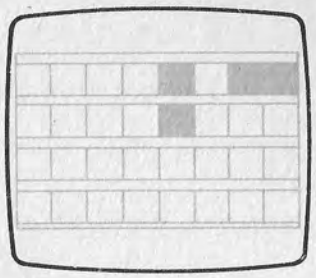


Fig. 3.

No hemos fallado ningún disparo, intenta conseguir el mismo promedio tú solo.

Cuando te sepas de memoria las posiciones correctas, modifica las posiciones y el número de cuadrados.

Definición de variables

Veamos qué ocurre cuando a la hora de definir las variables lo hacemos de distintas maneras y qué resultados obtenemos a la hora de ejecutar los procedimientos que las contienen.

El Logo nos permite dar dentro de un mismo procedimiento nombres iguales a las variables:

```
? PARA EJEMPLO :V :V :V
> AV :V
> FIN
```

A la hora de ejecutarlo tenemos que asignar tantos valores como variables tenemos y no bastaría con dar un solo valor, aunque las tres variables se denominen igual.

Si damos valores distintos a las variables:

```
? EJEMPLO 10 20 30
```

el valor que pasa al procedimiento es el primero que se ha tecleado, en este caso, 10.

El Logo también nos permite usar como nombre de variables el de las primitivas:

```
? PARA EJEMPLO :AV
> AV :AV
> FIN

? PARA EJEMPLO :AV :GD
> GD :GD
> AV :AV
> FIN
```

El nombre de la variable puede ser tan largo como queramos y puede estar compuesto de letras, números o signos, a excepción de:

- Los operadores aritméticos +, -, /, y *.
- Los operadores de relación: >, < y =
- Los paréntesis: ( )
- Los corchetes: [ ]

Si una expresión aritmética contiene nombres de variables, éstas deben contener objetos numéricos.

Un nombre de variable puede ser, por ejemplo:

```
? PARA EJEMPLO :V1J?:R"3
> AV :V1J2?:R"3
> FIN
```

Los caracteres que componen el nombre de variable no pueden estar separados por un espacio en blanco. Sería motivo de error definir una variable de la siguiente forma:

```
? PARA EJEMPLO :S 2
```

En este caso, recibimos un mensaje en el que nos dice que la orden PARA no acepta 2 como dato.

Si el espacio se deja al principio del nombre, ocurriría lo mismo que en el caso anterior:

```
? PARA EJEMPLO : S2
```

En este caso nos devuelve un mensaje diciéndonos que S2 no se acepta como dato.

Por último, podemos indicar que se va a utilizar una variable, pero no darle un nombre, basta con indicar los dos puntos después del nombre del procedimiento. En el lugar donde queremos utilizar la variable tenemos que indicarlo también con dos puntos exclusivamente:

```
? PARA EJEMPLO :
> AV :
> FIN
```

La variable en este caso no recibe un nombre, pero está identificada por el espacio en blanco.

A la hora de ejecutarlo, tenemos que dar un valor a la variable, como siempre hemos hecho:

```
? EJEMPLO 3
```

Asignaciones

En LOGO a todo aquello que no sea una acción se le denomina objeto. Un dato es un objeto LOGO; pueden ser números, caracteres, variables...

Una asignación es una operación a través de la cual damos un valor a una variable. En LOGO escribimos una asignación de la siguiente forma:

— HAZ "nombre de la variable "objeto

en este caso el objeto que asignamos a la variable es uno o varios caracteres, ya que hemos escrito delante del objeto unas comillas.

```
? HAZ "A "LOGO
```

hemos asignado la palabra LOGO a la variable A.

— HAZ "nombre de la variable objeto

en este otro caso el objeto sería un número.

```
? HAZ "B 1
```

asignamos a la variable B el valor 1.

Si ahora tecleas:

```
? ESCRIBE :A
LOGO

? ESCRIBE :B
1
```

observarás que, efectivamente, las variables A y B contienen los valores que les habíamos asignado.

Podemos modificar el contenido de una variable simplemente asignándole otro valor distinto:

```
? HAZ "A 20

? ESCRIBE :A
20
```

La variable A ya no contiene la palabra LOGO, ahora contiene el número 20.

Hemos visto que el objeto que asignamos a una variable puede ser un número o una palabra; veamos ahora que también puede ser el contenido de otra variable:

```
? HAZ "A 20

? HAZ "B :A
```

Primero le damos a la variable A el valor 20, luego le asignamos a B la variable A, es decir, el contenido de A; por tanto, el valor de B es también 20.

```
? ESCRIBE :A ESCRIBE :B
20
20
```

Podemos asignarle a una variable el contenido de otra.

Otra forma de realizar asignaciones es con la orden ASIGNA.

? ASIGNA objeto "nombre de variable

lo único que varía de una orden a otra es el orden de colocación del objeto y el nombre de la variable:

```
? ASIGNA 450 "A
? ESCRIBE :A
450
? ASIGNA "HOLA "A
? ESCRIBE :A
HOLA
```

NOTA: En el LOGO del SPECTRUM para realizar asignaciones sólo disponemos de la orden ASIGNA.

Familiarízate con esta nueva orden realizando unas cuantas asignaciones y observando los resultados:

```
? HAZ "A 1
? ESCRIBE :A
1
? HAZ "A 2
? ESCRIBE :A
2
```

hemos borrado el contenido anterior de A:

```
? HAZ "A SUMA 3 5
? ESCRIBE :A
8
```

asignamos a A el resultado de una suma:

```
? HAZ "A "LOLA
? ESCRIBE :A
LOLA
```

observa que podemos asignar a una misma variable números o letras indistintamente:

```
? ESCRIBE 100 + :A
+ NO ACEPTA LOLA COMO DATO
```

para poder realizar operaciones aritméticas con las variables éstas deben contener valores numéricos:

```
? HAZ "A 20
? ESCRIBE 100 / :A
5
? HAZ "A 10
? HAZ "B :A
? ESCRIBE :A
? ESCRIBE :B
10
10
```

hemos asignado a B el contenido de A, sin alterar a esta última:

```
? HAZ "A 40
? ESCRIBE :A
40
? ESCRIBE :B
10
? HAZ "A 50
? HAZ "B 100
? HAZ "C :A + :B
? ESCRIBE :C
150
```

hemos asignado a la variable C la suma de las variables A y B:

```
? ASIGNA 200 "A
? ASIGNA 10 "B
? ASIGNA RESTO :A :B "C
? ESCRIBE :C
0
```

Asignamos a C el resto de la división de A y B.

Cuadro resumen

— HAZ "Nombre de Variable "objeto
— HAZ "Nombre de Variable objeto

Se le asigna a la variable especificada un objeto que puede ser:

— un conjunto de caracteres, y hay que indicarlo con comillas.
— un valor numérico, y hay que indicarlo sin comillas.

— ASIGNA "objeto "Nombre de variable
— ASIGNA objeto "Nombre de variable

Realiza la misma operación que la orden HAZ, pero a la inversa.

Una misma variable puede contener números o caracteres indistintamente.

Ejercicios

1. *Dibuja un semáforo de tres discos y con un procedimiento de tres entradas coloréalos.*
2. *Intenta calcular los cuadrados de los 50 primeros números naturales.*
3. *Intenta realizar el siguiente dibujo:*

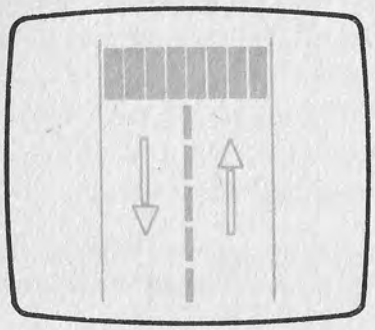


Fig. 4.

4. *Con los procedimientos que necesites y de las entradas necesarias, realiza una proceso para poder situar el número de sillas que queramos a cada lado de una mesa que puede ser de diferentes tamaños.*
5. *¿Son correctas las órdenes siguientes?*

```
? HAZ "A
? ASIGNA "A 40
? HAZ "AV 450
? PARA CASA :V 1
? PARA TREN :REPITE
```

Solución a los ejercicios

1:

Primero definimos el procedimiento para dibujar el semáforo con los tres discos.

```
? PARA SEMA
> PONCL 1
> SL
> PONX -16
> BL
> REPITE 2 [ AV 84 GD 90 AV 32 GD
  90 ]
> GD 90 AV 19
> REPITE 2 [ GD 90 AV 80 GD 90 AV
  16 ]
> SL PONPOS [ 12 16 ]
> PONRUMBO 180
> BL
> REPITE 36 [ AV 2 GD 10 ]
> SL PONPOS [ 12 43 ]
> BL
> REPITE 36 [ AV 2 GD 10 ]
> SL PONPOS [ 12 70 ]
> BL
> REPITE 36 [ AV 2 GD 10 ]
> FIN
```

Luego el procedimiento de tres entradas para colorear los discos.

```
:A color disco superior
:B color disco central
:C color disco inferior

? PARA COLOR :A :B :C
> SL
> PONPOS [ 5 70 ]
> BL
> PONCL :A
> RELLENA
> SL
> PONY 43
> BL
> PONCL :B
> RELLENA
> SL
> PONY 16
> PONCL :C
> BL
> RELLENA
> FIN
```

Por último, el procedimiento DIBUJO que inicializa la pantalla y realiza una simulación de un semáforo real cuando pasa de rojo a verde.

```
? PARA DIBUJO
> BP
> OT
> SEMA
> PONPALETA 0
> COLOR 2 0 0
> ESPERA 40
> COLOR 0 3 0
> ESPERA 40
> COLOR 0 0 1
> FIN
```

Las variables que se utilizan con la orden HAZ y ASIGNA reciben el nombre de variables globales.

Si tu LOGO no dispone de la orden PONPALETA, no la teclees.
Si los códigos de colores no corresponden a:

1 verde
2 rojo
3 ámbar

sustituye en el procedimiento estos códigos por los que utilice tu ordenador.
Ejecuta el dibujo tecleando

? DIBUJO

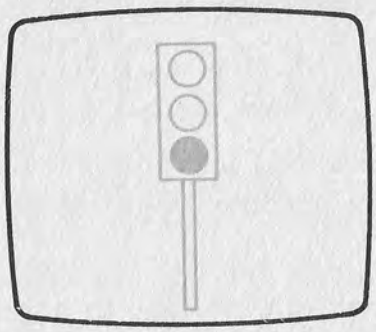


Fig. 5.

2:

```
? PARA CUADRADOS
> PT
> ESCRIBE "CUADRADOS
> HAZ "B 0
> REPITE 50 [ HAZ "C :B + 1 HAZ "C2
  :C * :C ESCRIBE :C2
  HAZ "B :C ESPERA 10 ]
> FIN
```

Habrás pensado que es una forma un tanto rebuscada de resolver el ejercico, y así es. Más adelante estudiaremos otras maneras de resolverlo mucho más sencillas; eso será cuando conozcamos un poco mejor el LOGO.
El método que hemos seguido es el siguiente:

— Inicializamos la variable B a cero.
— Repetimos 50 veces el proceso:

- Asignar a C el valor de B más uno.
  La primera vez C será igual a 1.
  La segunda vez C será igual a 2, pues B ya valdrá 1.
  Así sucesivamente en C tendremos (1, 2, 3, 4, ..., 50) todos los números naturales hasta el 50.
- Elevar al cuadrado C. Lo hemos hecho multiplicando C por sí misma porque no todas las versiones de LOGO disponen de la función POTENCIA a b. El resultado lo almacenamos en la variable C2 y luego la escribimos.
- Asignar a B el contenido de C.
  Con esto conseguimos que B se incremente en 1.
  La primera vez B tendrá el valor 1.
  La segunda vez el valor 2, pues C ya valdrá 2.

Puedes variar el límite de los 50 primeros cambiando el 50 del REPITE por el número que quieras.
Ejecuta el ejercicio tecleando:

? CUADRADOS

3:

Definimos un primer procedimiento para dibujar los márgenes de la carretera y las dimensiones:

```
? PARA MARGEN
> SL
> PONPOS [ -50 -50 ]
> BL
> AV 160
> SL
> PONPOS [ 50 -50 ]
> BL
> AV 160
> FIN
```

Con el siguiente dibujamos la línea discontinua de separación de carriles:

```
? PARA LINEA
> SL
> PONPOS [ -2 -50 ]
> BL
> REPITE 5 [ REPITE 2 [ BL AV 20 GD
  90 AV 4 GD 90 ] SL AV 25 ]
> SL
> PONPOS [ -1 -40 ]
> REPITE 5 [ BL RELLENA SL AV 25 ]
> FIN
```

A continuación el que dibuja el paso de cebra:

```
? PARA CEBRA
> SL
> PONPOS [ -47 75 ]
> REPITE 8 [ REPITE 2 [ BL AV 30 GD
```

Un procedimiento puede tener un número de entradas superior a dos.

```
    90 AV 10 GD 90 ] GD 90 SL AV 12 GI
    90 ]
> SL
> AV 15 GI 90 AV 7
> REPITE 8 [ BL RELLENA SL AV 12 ]
> FIN
```

Y, por último, el que dibuja las flechas que indican el sentido de la circulación en los dos carriles:

```
? PARA FLECHAS
> SL
> PONPOS [ 23 -10 ]
> PONRUMBO 0
> BL
> REPITE 2 [ AV 40 GD 90 AV 4 GD 90
  ]
> AV 41 GI 90 AV 5
> PONPOS [ 25 50 ]
> PONPOS [ 32 31 ]
> PONX 24
> SL
> PONPOS [ -23 50 ]
> PONRUMBO 180
> BL
> REPITE 2 [ AV 40 GD 90 AV 4 GD 90
  ]
> AV 41 GI 90 AV 5
> PONPOS [ -25 -10 ]
> PONPOS [ -32 9 ]
> PONX -24
> FIN
```

Una vez que tenemos todos los procedimientos definidos por separado, los agrupamos en uno solo:

```
? PARA CARRETERA
> BP
> OT
> MARGEN
> LINEA
> CEBRA
> FLECHAS
> FIN
```

Para ejecutarlo

```
? CARRETERA
```

4:

Definimos un procedimiento para dibujar la mesa, donde tanto el largo como el ancho pueden ser variables. Centramos la mesa utilizando las variables que nos determinan las dimensiones de la mesa, situando la Tortuga a una distancia del centro y a la izquierda igual a la mitad del largo y a una distancia del centro y hacia abajo igual a la mitad del ancho:

```
? PARA MESA :LARGO :ANCHO
> SL
> PONX -:LARGO / 2
> PONY -:ANCHO / 2
> BL
> REPITE 2 [ AV :ANCHO GD 90 AV
  :LARGO GD 90 ]
> SL
> PONX COORX + 5
> PONY COORY + 5
> BL
> REPITE 2 [ AV :ANCHO - 10 GD 90
  AV :LARGO - 10 GD 90 ]
> FIN
```

A continuación definimos un procedimiento para dibujar las sillas. Estas van a ser todas de igual tamaño y van a tener una anchura de 18 puntos. Una vez que dibuja una, la Tortuga se sitúa en la posición donde tiene que comenzar a dibujar la siguiente:

```
? PARA SILLA
> BL
> REPITE 4 [ AV 10 GD 90 ]
> AV 2 GI 90
> REPITE 2 [ BL AV 2 GD 90 AV 6 GD
  90 AV 2 SL AV 10 ]
> SL
> PONRUMBO 0
> RE 2
> PONX COORX + 18
> FIN
```

Con el procedimiento LATERAL1, dibujamos el número de sillas que determina la variable N:

```
? PARA LATERAL1 :N
> SL
> PONX (-:N * 18 / 2) + 4
> PONY (:ANCHO / 2) + 5
> REPITE :N [ SILLA ]
> FIN
```

Calculamos los puntos que ocupan todas las sillas para centrarlas y posicionamos a la Tortuga donde tiene que empezar a dibujarlas.

Podemos dar el nombre de una primitiva del Logo a una variable.

Con LATERAL2, hacemos lo mismo que con LATERAL1, pero en la parte inferior:

```
? PARA LATERAL2 :N1
> SL
> PONX (-:N1 * 18 / 2) + 4
> PONY (-:ANCHO / 2) - 15
> REPITE :N1 [ SILLA ]
> FIN
```

Tanto en LATERAL1 como en LATERAL2 se repite un número determinado de veces una llamada al procedimiento SILLA.
Una vez que tenemos todos los procedimientos, definimos uno que contenga una llamada a todos ellos y por el que pasamos los valores a todos ellos:

```
? PARA SALA :LARGO :ANCHO :N
  :N1
> BP
> OT
> MESA :LARGO :ANCHO
> LATERAL1 :N
> LATERAL2 :N1
> FIN
```

Si queremos dibujar una mesa de 100 × 30, con 5 sillas en la parte superior y 2 en la inferior:

```
? SALA 100 30 5 2
```

y obtenemos:

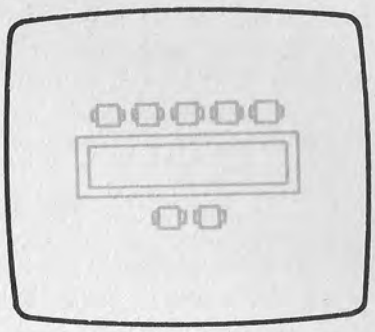


Fig. 6.

Para una mesa de 200 × 40, con 10 sillas en la parte superior y 5 en la inferior:

```
? SALA 200 40 10 5
```

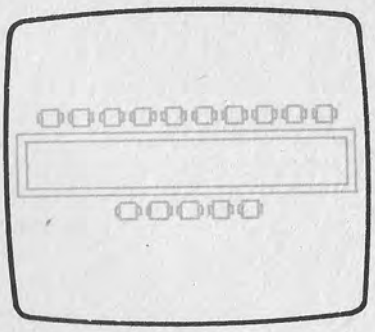


Fig. 7.

De la siguiente forma, obtenemos una mesa cuadrada de 50 × 50 con 3 sillas a cada lado:

```
? SALA 50 50 3 3
```

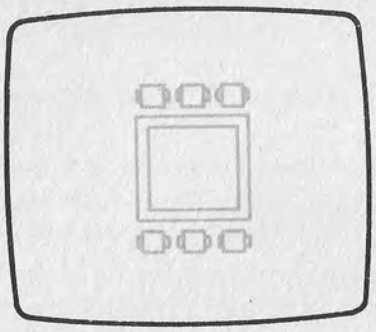


Fig. 8.

5:

```
—? HAZ "A
```

INCORRECTO. No hemos indicado el objeto que queremos asignar a la variable a.

LOGO muestra el mensaje:
FALTAN DATOS PARA HAZ

```
—? ASIGNA "A 40
```

INCORRECTO. Si utilizamos la orden ASIGNA debemos invertir el orden de colocación, primero el objeto y luego la variable.

```
—? HAZ "AV 450
```

CORRECTO. Asignamos a la variable AV el valor 450.

```
—? PARA CASA :V 1
```

INCORRECTO. No se pueden dejar espacios en blanco entre los caracteres que forman el nombre de una variable.

```
—? PARA TREN :REPITE
```

CORRECTO. EL LOGO nos permite utilizar como nombre de variables el de las primitivas.

El nombre de una variable puede contener un gran número de caracteres.

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

OMO prometimos en el tomo anterior, en éste vamos a ver la explicación del programa EL GUSANO LOCO. Espero que todos hayan estudiado el funcionamiento de dicho programa. En caso afirmativo será más fácil comprender la explicación que aparece a continuación. Como siempre, veremos el funcionamiento del programa línea a línea.

Línea 180. Borramos la pantalla.

Líneas 190 a 270. Se imprimen en pantalla las teclas que están permitidas utilizar para mover el gusano por la pantalla. Como se puede apreciar por el programa, las teclas no se pueden definir. Esto se hizo así para que el programa ocupase menos espacio y para no asustar al usuario a la hora de introducirlo en el ordenador. Invito a todos los lectores a realizar las modificaciones necesarias en este programa para que acepte la redefinición de teclas por parte del usuario.

Línea 280. Dejamos dos líneas en blanco.

Línea 290. Preguntamos que con qué nivel de dificultad quiere jugar el usuario. En el nivel cero el gusano se moverá muy de prisa. En el nivel cuatro, éste se moverá bastante despacio.

Línea 300. Si la tecla que ha pulsado el usuario tiene un código ASCII menor que cero (0) o mayor que cuatro (4), entonces el programa repite la pregunta sobre el nivel del juego.

Línea 310. Calculamos el retardo de tiempo que realizará el programa dependiendo del nivel elegido por el usuario. Si éste eligió un nivel cero (0), el retardo será nulo. Si eligió un nivel cuatro (4), éste será de 100 pasos.

Línea 370. Volvemos a borrar la pantalla.

Línea 380. Se le dice al usuario que espere un momento.

Línea 390. Dimensionamos todos los vectores que van a ser necesarios durante el transcurso del juego. Estos son:

— A$() — Almacenará los hombres de las personas que han conseguido un récord.

— S() — Almacenará la puntuación de las personas que han conseguido un récord.

— F$() — Almacenará las distintas frutas que se puede comer el gusano.

Línea 400. En esta línea comienza un bucle dentro del cual vamos a asignar a todos los elementos del vector A$() el nombre de Siglo Cultural. Como recordaréis, este vector se encarga de almacenar los nombres de las personas que han conseguido una puntuación más alta. Como al principio del programa no ha jugado nadie todavía, entonces podemos asignar a todos los elementos del vector el nombre de la editorial.

Línea 410. Se asigna a cada elemento del vector A$() el nombre de la editorial Siglo Cultural.

Línea 420. Se almacena en todos los elementos del vector numérico S() la puntuación 100.

Línea 430. Aquí se termina el bucle.

Línea 440. Se asigna a la variable NV el número 1 y a la variable VV el número 3. La variable NV nos servirá para saber en qué número de pantalla se encuentra el usuario. La variable VV nos dice el número de vidas del jugador. Si quieres, y eres muy malo jugando, puedes cambiar el valor de VV para tener diez vidas o más.

Línea 450. Dimensionamos una tabla de 40 por 23 elementos en la cual almacenaremos, como si de la pantalla se tratase, toda la información que tenemos sobre el movimiento y estado del gusano. Los números que iremos almacenando serán los siguientes según su significado:

— 0 — En dicha posición de la pantalla no hay nada.

— 1 — En dicha posición de la pantalla hay un ladrillo con el que se puede chocar el gusano.

— -1 — En dicha posición de la pantalla se encuentra parte del cuerpo del gusano.

— 2 — En dicha posición de la pantalla se encuentra la fruta que hemos definido como '1'.

— 3 — En dicha posición de la pantalla se encuentra la fruta que hemos definido como '%'.

— 4 — En dicha posición de la pantalla se encuentra la fruta que hemos definido como '$'.

— 5 — En dicha posición de la pantalla se encuentra la fruta que hemos definido como '='.

Gracias a esta matriz no tenemos que estar mirando continuamente en la pantalla, sino que, con sólo mirar a dicha matriz, podemos ver lo que hay en la pantalla. Todo lo que vayamos imprimiendo en pantalla lo iremos almacenando en esta matriz (A(x,y)).

Línea 460. Definimos la primera de las frutas. Esta es la que tiene la forma de almohadilla (ı). Su valor durante el juego será de 30 puntos.

Línea 470. Definimos otra de las frutas que interviene en el juego. Esta tendrá forma de tanto por ciento (%). Su valor será de 45 puntos.

Línea 480. En esta línea definimos la siguiente fruta. Tendrá forma de dólar ($) y su valor será de 60 puntos.

Línea 490. Almacenamos en el último elemento del vector F$() el carácter que actuará como la fruta que nos dará más puntuación. Este será el signo igual (=). Su valor será de 75 puntos.

Líneas 550 a 590. Realizamos un bucle, dentro del cual ponemos a cero todos los elementos de la matriz A(). Cuando el programa se ejecuta por primera vez esto no es necesario, ya que el ordenador lo hace automáticamente, pero, una vez que hemos superado la primera pantalla, al igual que borramos la pantalla, tenemos que borrar todos los elementos de la matriz.

Línea 600. Definimos la variable LO con valor cero (0). Esta variable nos indicará el número de caracteres que ha crecido el gusano después de comerse una fruta.

Línea 610. Borramos la pantalla.

Líneas 620 a 640. Inicializamos una serie de variables con valor diez (10). Estas variables tienen la siguiente función:

X e Y — Nos indicarán la posición de la pantalla donde tenemos que imprimir, a continuación, la cabeza del gusano.

X1 e Y1 — Nos indicarán la última posición de la cabeza del gusano para sustituirla por parte del cuerpo.

X3 e Y3 — Nos indicarán la posición de la cola del gusano. Gracias a esta variable podremos borrar la cola del gusano según éste vaya avanzando.

Línea 650. Hacemos una llamada a la **línea 710.** En esta línea nos encargaremos de dibujar los límites de la pantalla. Los límites del recinto sobre el que se moverá el gusano.

Línea 660. Según el valor que tenga la variable numérica NV, iremos a una de las siguientes líneas:

Valor 1 — Línea 900
Valor 2 — Línea 960
Valor 3 — Línea 1100
Valor 4 — Línea 1360

Cada una de estas líneas corresponde a una pantalla distinta. Cada vez que el jugador haya superado una de las pantallas, el valor de NV se incrementará y, por tanto, se continuará el juego en otra pantalla distinta.

Línea 670. Se hace una llamada a la rutina encargada de poner las frutas en la pantalla.

Línea 680. Llamamos a la rutina que se encuentra a partir de la **línea 2040** y que se encarga del movimiento del gusano.

Línea 690. Aquí se realiza un bucle de retardo que es el que controla el tiempo que tarda en moverse el gusano entre posición y posición. Este bucle es el que define el jugador al elegir el nivel de juego.

Línea 700. Volvemos a irnos a la **línea 670** para continuar con el juego.

Hasta aquí podemos decir que llega el programa principal. A partir de esta línea empiezan a aparecer todas las rutinas que hacen posible el juego. Unas se irán llamando a otras.

Entre la **línea 760 y 880** nos dedicamos

a dibujar los límites de la pantalla o, mejor dicho, los límites del campo sobre el cual se podrá mover el gusano. La forma de hacerlo es muy sencilla. Lo único que hay que hacer es imprimir un rectángulo en la pantalla y que coincida con el máximo rectángulo que se puede realizar dentro de la misma.

En la **línea 950** sólo hay un RETURN, porque la primera pantalla es igual a los límites del campo sobre el que se moverá el gusano.

Las líneas comprendidas entre la **1010** y la **1080** nos permitirán dibujar, aparte de los límites del campo, dos barras verticales, que actuarán como dos muros, con lo que el movimiento del gusano se complicará un poco. Esta es la segunda pantalla que tendrá que soportar el jugador si sobrepasa la primera.

Las líneas del programa que van desde la **1150** y la **1340** nos imprimirá la tercera pantalla de las que consta el juego. Esta pantalla es igual que la primera, pero con una subdivisión del campo en el centro de la pantalla. Dicho campo tendrá dos puertas (una a la derecha y otra a la izquierda) para que el gusano pueda entrar y salir como quiera.

El último grupo de líneas que se encarga de imprimir pantalla se encuentra a partir de la **línea 1410** y hasta la **1480.** Esta cuarta pantalla es igual que la tercera, pero con la diferencia de que en mitad del cuadrado que se encuentra en medio del campo se imprime otra nueva muralla, esta vez maciza, para hacer más difícil el movimiento del gusano. Como se puede apreciar en esta rutina, primero se llama a la rutina encargada de dibujar la pantalla número tres y después se imprime el nuevo muro.

A continuación vienen el grupo de líneas que nos avisa que nos hemos chocado contra la pared:

Línea 1560. Se advierte al jugador que se ha chocado con algunas de las paredes que hay en la pantalla.

Línea 1570. Y le decimos que la próxima vez tenga más cuidado.

Línea 1580. Si el número de vidas (VV) era igual a uno (1), o sea, que ya no tiene ninguna, se manda el control del programa a la **línea 2450.** En esta línea se imprime por pantalla la tabla de ganadores.

Línea 1590. Se resta uno al número de vidas.

Línea 1600. Devolvemos el control de la **línea 550.** Hay que recordar que, a partir de esta línea, se ponen a cero todos los elementos de la matriz A().

Ahora viene el grupo de líneas que nos advierte que nos hemos chocado contra nosotros mismos:

Línea 1670. Se le dice al jugador que se ha chocado consigo mismo.

Línea 1680. Se le manda un mensaje para que la próxima vez tengas más cuidado.

Línea 1690. Si el número de vidas es igual a uno (y el jugador acaba de desperdiciar esta última vida) nos vamos a las subrutinas de impresión de récords del día.

Línea 1700. Le restamos uno al número de vidas.

Línea 1710. Y nos volvemos a la **línea 550**. A partir de esta línea, como ya hemos visto, volvemos a poner la variable numérica A(), y que controla todo lo que hay impreso en pantalla, a cero.

En el grupo de líneas que aparecen a continuación veremos cómo el programa se encarga de colocar, aleatoriamente, las frutas por la pantalla.

Línea 1770. Sacamos un número aleatorio entre 0 y 1. Si este número es mayor que 0.1, entonces devolvemos el control al programa principal. Aunque parezca que muy pocas veces aparecerá un número menor de 0.1, las veces que aparezca serán suficientes para las frutas que queremos colocar en la pantalla. Incluso habrá veces en que el número de frutas sea mayor del deseado. En tal caso se puede cambiar esta línea por:

```
1770 IF RND.085 THEN RETURN
RND > .085
```

Línea 1780. En el caso de que el número que nos ha salido sea menor de 0.1, entonces sacamos cuál de las cuatro frutas va a aparecer en la pantalla.

Línea 1790. Sacamos la posición, en X, de la pantalla donde se colocará dicha fruta.

Línea 1800. Así como la posición en Y.

Línea 1810. Miramos si dicha posición ya está ocupada por un muro, por el gusano o por otra fruta. En caso afirmativo, nos volvemos al programa general. Podíamos habernos ido a la **línea 1790,** pero con esto conseguimos que el número de frutas en la pantalla se reduzca y que el programa no se vuelva lento.

Línea 1820. Almacenamos en la matriz A() el valor de dicha fruta más uno (mírese un poco más arriba la razón de esto).

Línea 1830. Colocamos el cursor en la posición que hemos hallado en las **líneas 1790**

y 1800 e imprimimos la fruta que elegimos en la **línea 1780.**

Línea 1840. Devolvemos el control al programa principal.

El siguiente grupo de líneas se encarga de mirar si el gusano ha chocado con algo. El gusano puede chocar:

— Consigo mismo.
— Con alguna de las paredes.
— Con una de las frutas.

Esta parte del programa no sólo mira si el gusano se ha chocado, sino, también, contra qué ha chocado:

Línea 1900. Preguntamos, mirando en la matriz A(), si en la siguiente posición que tendrá el gusano hay algo. En caso negativo, devolvemos el control al programa principal.

Línea 1910. Si en dicha posición nos encontramos que hay un menos uno (-1), entonces significa que nos hemos chocado contra nosotros mismos. En ese caso nos vamos a la **línea 1610,** que ya hemos explicado.

Línea 1920. Si lo que nos encontramos es un uno (1), entonces es que hemos chocado contra una pared. En ese caso transferimos el control del programa a la **línea 1500.**

Línea 1930. Si en dicha posición no había pared y tampoco había parte del cuerpo del gusano, significa que nos encontramos ante una fruta. Entonces decimos que los puntos acumulados hasta el momento (SC) sean los que teníamos antes más 15 multiplicado por el valor de la matriz A() en dicho punto.

```
LET SC=SC+15*A(X,Y)
```

También asignamos a la variable numérica LO el número de segmentos que crecerá el gusano. Este número será proporcional a la fruta comida por el gusano.

Línea 1940. Se pregunta si el número de frutas que se ha comido el usuario es distinto de quince (15). En caso afirmativo se suma uno al acumulador que se encarga de sumar el número de frutas comidas y se devuelve el control al programa principal. Si se quiere que el juego sea más complicado, sólo hay que cambiar el número máximo de frutas comidas por pantalla.

Línea 1950. En el caso de que el número de frutas sea igual a 15, entonces borramos la pantalla y hacemos que el número de frutas comidas sea igual a cero (0), para estar preparados para entrar en el siguiente nivel.

En el grupo de líneas que aparece a continuación se le dice al jugador que ha terminado con la pantalla. Dependiendo del número de pantalla en la cual se encontrase, se pasará a la siguiente (y se avisará al jugador) o se terminará el juego. Este termina cuando el jugador ha terminado las cuatro pantallas.

En la **línea 2020** se realiza un bucle de retardo y en la **2030** se manda el control del programa a la **línea 2450.** En esta línea se imprime la tabla de récord del día.

El grupo de líneas que aparecen a continuación no tienen ninguna dificultad, salvo las últimas. El lector ha de estar preparado para conocer el funcionamiento del grupo de líneas que va desde la **2090** hasta la **2200.** El resto de las líneas serán analizadas a continuación:

Línea 2210. Como acabamos de imprimir en pantalla una nueva posición del gusano, entonces tenemos que almacenar en la matriz A() un menos uno en dicha posición.

Línea 2220. Preguntamos si LO es igual a cero. Esta variable es igual a cero cuando no nos hemos comido ninguna fruta o cuando, después de habernos comido una, ya hemos crecido lo suficiente. En el caso de que dicha variable sea igual a cero, eso significa que la cola del gusano se tiene que mover. Para hacer esto nos vamos a la **línea 2250.**

Línea 2230. Si dicha variable es distinta de cero, le restamos uno a su valor. Esto significa que el gusano tiene que seguir creciendo.

Línea 2240. Devolvemos el control al programa principal.

Línea 2250. Colocamos el cursor en la posición de la cola del gusano.

Línea 2260. Y borramos el último carácter.

Línea 2270. Como hemos borrado dicha posición de la pantalla, también tenemos que borrarla de la matriz A(). Así lo hacemos.

Líneas 2280 a 2310. En este grupo de líneas se mira por dónde continúa la cola del gusano. Una vez que se encuentra continuación se devuelve el control al programa principal. En este grupo de líneas se coge como continuación la parte del gusano que primero se vea. Con esto se consigue que, a veces, trozos del gusano permanezcan en pantalla, haciendo más difícil el juego, ya que no se sigue exactamente el movimiento del gusano.

Línea 2320. Devolvemos el control al programa principal.

El grupo de líneas entre la **2380** y la **2440** se encargan de preguntarle al usuario si

desea jugar una nueva partida. Debido a la sencillez de esta parte del programa, no se explica.

A continuación viene la parte del programa que se encarga de almacenar, imprimir y pedir por pantalla los récords de los jugadores que han jugado hasta el momento. Esta parte del programa, por ser muy importante en todos los programas de juegos, la veremos en sucesivos tomos. En ellos observaremos cómo construirla y qué características tiene que tener.

Por fin, el último grupo de líneas es el encargado de decir adiós al jugador; se encuentran a partir de la **línea 2960** y hasta el final. Tampoco es necesario explicar su funcionamiento, pues es de todos conocidos.

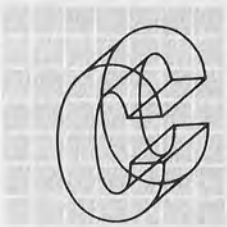
Una vez que hemos visto cómo funciona el programa, nos podemos dar cuenta de que hacer un juego es algo muy sencillo. Pero esta sencillez se puede complicar casi hasta el infinito, ya que siempre se pueden añadir gráficos (en este caso habría que hacerlos desde el principio), aumentar el número de pantallas (por qué no lo intentáis), aumentar el grado de dificultad...

Una de las modificaciones que se pueden realizar en este programa sin tener que pensar mucho es la siguiente. Se podría intentar que, de tanto en tanto apareciesen por la pantalla una serie de frutas que, en vez de aumentar nuestra longitud, nos hiciese un poco más pequeños.

Otras variaciones que os recomiendo que intentéis son:

1. En vez de cambiar de pantalla cada 15 frutas comidas, se puede modificar el programa para que cambie al transcurrir un cierto tiempo.
2. Se pueden añadir más pantallas al juego.
3. Puede haber otro gusano moviéndose por la pantalla y con el cual tenemos que intentar no chocar.

Como podéis ver, las modificaciones que se le pueden hacer a un programa son casi infinitas. Todo depende de uno mismo. Por eso mismo, si queréis realizar juegos que sean buenos en vuestro ordenador, es conveniente que intentes practicar tú mismo.



COMO prometimos en el tomo anterior, en éste vamos a ver el funcionamiento del programa GESTOR DE FICHEROS, que apareció en el tomo 11. Como durante una serie de tomos hemos ido explicando el funcionamiento de algunas rutinas, sólo veremos cómo funciona el programa principal y todas aquellas rutinas que no hayamos visto o que aparecen con alguna modificación.

Línea 220. En esta línea definimos una serie de vectores, numéricos y alfanuméricos, que serán los encargados de almacenar toda la información sobre el fichero. Estos vectores son:

— F$(). En él almacenaremos todas las fichas del fichero. Cada elemento del vector se corresponde a un registro del fichero.

— A$(). En este vector almacenaremos las distintas opciones de los menús que tiene el programa.

— N(). Nos servirá para saber la definición del fichero.

— N$(). Esta es una matriz de dos dimensiones. En ella almacenaremos todas las características de la impresión. Almacenaremos el formato de impresión.

Por supuesto, los índices de todos estos vectores se pueden variar dependiendo de las necesidades de cada usuario. Por ejemplo, si necesitamos un fichero en el que cada ficha pueda tener mil caracteres, entonces la matriz F$() no podrá tener mil elementos, puesto que no habría memoria suficiente. En tal caso se recomienda cambiar la dimensión de los vectores cuando sea necesario.

Línea 230. Definimos la variable TT con valor cero. Esta variable será la que nos indicará el número de fichas en memoria que tiene el fichero.

Línea 240. Borramos la pantalla.

Línea 250. Definimos la variable A$ con un mensaje. Si se recuerda, a la rutina de creación de menús teníamos que pasarle una serie de parámetros. Uno de ellos es la cabecera de dicho menú.

Línea 260. Asignamos a N el número de opciones que tiene dicho menú.

Líneas 270 y 280. Asignamos a los dos primeros elementos de la matriz A$() las dos opciones del menú.

Líneas 290 y 300. Definimos cuál puede ser el carácter con código ASCII más alto que puede pulsar el usuario, y el que tiene código ASCII más bajo.

Línea 310. Llamamos a la subrutina de creación de menús. Como se recordará, esta rutina no sólo crea el menú, sino que también se encarga de recoger la opción del usuario y borrar la pantalla.

Línea 320. Si el usuario respondió con un uno (1), o lo que es igual, si el usuario quiere definir un fichero nuevo, nos vamos a la **línea 7250.** Esta rutina ya la vimos en tomos anteriores, por lo que no es necesario explicarla. Cuando el programa termina con esta rutina, y vuelve al programa general, se transfiere de nuevo el control del programa a la **línea 240.**

Línea 330. En el caso de que el usuario quiera entrar en el menú principal (pulsó un dos (2) en el menú general), borramos la pantalla.

Líneas 340 a 440. En este grupo de líneas se realiza la misma operación que en las

que hemos visto un poco más arriba. Lo que hacemos es definir la cabecera del menú principal, las opciones de éste, el número de opciones y el carácter con el código ASCII más alto permitido. Grupos de líneas como éste se repiten durante todo el programa cada vez que es necesario imprimir un menú por la pantalla.

Línea 450. Llamamos a la rutina generadora de menús para que me imprima el menú general.

Línea 460. Hacemos que la variable numérica A adquiera el valor de A$. Como A$ será un carácter comprendido entre el carácter 1 y el carácter 8, A tomará el valor de dicho carácter.

Línea 470. Según el valor de la variable A, nos iremos a una de las líneas que aparecen a continuación y cuya función es la siguiente:

490. A partir de esta línea se encuentra la parte del programa que nos permitirá introducir fichas nuevas en el fichero.

650. Nos vamos a la parte del programa que se encarga de gestionar las fichas. Esta gestión consiste en borrar una ficha, limpiar el fichero de fichas borradas, modificar una ficha y ver una ficha en particular.

1050. Aquí empieza la parte del programa que se encarga de ordenar las fichas.

1150. A partir de esta línea empieza la parte del programa que nos permitirá buscar una o más fichas. Podremos buscar fichas preguntando por el valor de un cierto campo, por su numeración o bien buscar, y ver, las fichas que hemos borrado en otro momento.

1450. Para todo lo relacionado con la grabación del fichero.

1990. Para leer un fichero.

2550. A partir de esta línea se encuentran las rutinas que nos permitirán definir la impresión e imprimir las fichas.

3200. Para terminar la sesión de hoy con el programa.

Línea 480. Una vez que el programa haya hecho lo que nosotros le hayamos dicho en cualquiera de las partes del programa que acabamos de ver, entonces nos volvemos al menú principal.

Hasta aquí se puede considerar que llega el cuerpo del programa principal. A partir de esta línea, nos vamos a encontrar las diversas partes del programa que se encargan de gestionar todas las funciones que es capaz de realizar.

Línea 520. Entramos de lleno en la parte que nos permitirá introducir fichas. Borramos la pantalla.

Línea 530 y 540. Sacamos un mensaje por pantalla para que el usuario sepa en qué parte del programa se encuentra.

Línea 550. Se saca un mensaje diciendo el número de fichas que tenemos en memoria.

Línea 560. Hacemos que NF (Número de Ficha) sea igual al número de fichas más uno. Esta variable es la que va a controlar el número de ficha que vamos a introducir. Por tanto, es lógico que su valor sea una unidad más alto que TT. En esta línea también definimos que la ficha NF es igual al carácter 254. Si nos acordamos, el carácter 254 iba a ser el encargado de separar los diferentes campos del registro.

Línea 570. Nos vamos a la **línea 8010.** En esta línea y en las siguientes, será donde, en realidad, introduzcamos las fichas.

Línea 580. Hacemos que TT (número de fichas en memoria) se incremente en uno.

Línea 590. Y colocamos el cursor en la **línea 20** para preguntar...

Línea 600. ... si deseamos introducir otra ficha.

Líneas 610 a 640. Se recoge del teclado la respuesta del usuario. Si ésta resultó afirmativa, nos volvemos a la **línea 490.** En caso contrario, devolvemos el control al programa general. Si el usuario no pulsó ni S ni N, entonces se vuelve a preguntar.

A partir de la **línea 680** se encuentra la parte del programa que se encarga de gestionar las fichas. En esta parte del programa, debido a que tenemos varias opciones posibles, necesitamos un menú. Así entre las **líneas 680** y **740** nos encargamos de fabricarlo, por así decirlo, y de mandar el control del programa a la rutina creadora de menús.

Línea 760. Según la opción elegida por el usuario, nos vamos a una línea o a otra.

Línea 770. A partir de esta línea nos encargaremos de borrar una ficha. En esta línea y en la siguiente sacamos un mensaje por pantalla para informar al usuario.

Línea 800. En el caso de que no haya fichas en el fichero (en el caso de que el fichero no esté definido), se imprime un fichero advirtiéndolo y se transfiere el control a la **línea 5440.** A partir de esta línea se encuentra la rutina encargada de esperar que el usuario pulse una tecla. Una vez que lo ha hecho, se devuelve el control al programa principal.

Línea 810. Se pregunta el número de ficha que se quiere borrar.

Línea 820. Si el número de ficha no está dentro de los límites permitidos, se vuelve a realizar la misma pregunta.

Línea 830. Y borramos la ficha. Se recordará que no se borraban realmente las fichas, sino que se les ponía el carácter 253 delante de su contenido como una bandera que nos indicase que dicha ficha estaba borrada.

Línea 840. Retornamos al programa principal.

Línea 850 y 860. Se advierte al usuario que vamos a limpiar el fichero de todas las fichas borradas.

Línea 870. Transferimos el control a la rutina limpiadora.

Línea 880. Y devolvemos el control al programa principal.

Líneas 890 a 910. Sacamos un mensaje por pantalla avisando al ordenador que estamos en la parte del programa encargada de modificar fichas.

Línea 920. Se pregunta qué ficha queremos modificar.

Línea 930. Si el número de ficha introducido no existe (no está dentro de los límites permitidos), se advierte al usuario que dicha ficha no existe, se imprime el mensaje de pulsa una tecla (GOSUB 5440) y se retorna al programa principal.

Línea 940. Nos vamos a la **línea 8290.** En esta línea y en las siguientes lo que el programa hace es imprimir por pantalla el contenido de dicha ficha.

Línea 950. Borramos la pantalla.

Línea 960. Y llamamos a la rutina de introducción de fichas para que el usuario modifique la ficha.

Línea 970. Devolvemos el control al programa principal.

Línea 980. A partir de esta línea empieza la rutina que nos permitirá ver el contenido de una ficha.

Línea 1010. Se pregunta por el número de ficha.

Línea 1020. Si la ficha no existe, entonces se avisa al usuario y se le pide que pulse una tecla (GOSUB 5440) y se devuelve el control al programa principal.

Línea 1030. Llamamos a la rutina que se encuentra a partir de la **línea 8290** y que se encargará de hacer visible dicha ficha.

Línea 1040. Devolvemos el control al programa principal.

Líneas 1090 a 1110. Nos encontramos en la parte del programa que se encarga de gestionar la ordenación de las fichas.

Línea 1120. Se le pregunta al usuario por qué campo quiere ordenar el fichero.

Línea 1130. Llamamos a la rutina ordenadora. Como recordaréis, es la propia rutina la que se encarga de mirar si el campo por el cual el usuario quiere ordenar existe o no.

Línea 1140. Y se devuelve el control al programa principal.

Las líneas que vienen a continuación se encargan de preparar todas las variables necesarias para la impresión del menú de búsqueda de fichas y de llamar a la subrutina de creación de menús.

Línea 1330. Dejamos dos líneas en blanco, después de decirle al usuario en qué parte del programa nos encontramos y hacemos que la variable SW adquiera el valor de la opción elegida por el usuario menos uno. Se recordará que SW es la variable que le dice a la rutina de búsqueda qué tipo de búsqueda quiere el usuario.

Línea 1340. Según la opción elegida por el usuario, nos vamos a una línea o a otra.

Entre las **líneas 1350** y **1380,** como el usuario quiere buscar por número de ficha, se pregunta desde qué ficha hasta qué ficha se va a realizar la búsqueda.

Entre las **líneas 1390** y **1420,** como el usuario quiere buscar las fichas que tengan un cierto valor en un cierto campo, se le pregunta qué valor y qué campo.

Por fin en la **línea 1430** se realiza una llamada a la rutina que se encarga de buscar las fichas. Dentro de esta rutina, que ya hemos explicado, no sólo se buscan las fichas, sino que también se llama a la rutina encargada de imprimirlas.

A partir de la **línea 1450** están localizadas las rutinas encargadas de grabar el fichero. Primero se creará un submenú en el cual se pregunta al usuario si la grabación va a ser en disco o en cinta. Según la opción escogida, el ordenador saltará a la **línea 1640** o a la **1810.** Por supuesto, antes de ir a ninguna de estas líneas, se pregunta con qué nombre se grabará el fichero.

Como ambos grupos de líneas son muy parecidos, la única diferencia es el canal por el cual se graba, explicaremos sólo uno de ellos. La otra opción de grabación es exactamente igual.

Línea 1640. Se abre el fichero con el nombre que ha dado el usuario. En el SPEC-

TRUM, como no es necesario abrir un fichero para grabar en él, se graba directamente sobre el vector N().

Línea 1650. Primero grabaremos la configuración del fichero. Así, en esta línea grabamos el número de campos de cada ficha. En el programa que tienen los usuarios del SPECTRUM, en esta línea se graban los nombres de cada campo.

Línea 1660. A partir de esta línea comienza un bucle, denro del cual se va a grabar la configuración del fichero. En el SPECTRUM, esta línea se encarga de grabar el contenido del fichero.

Línea 1670. Se graban uno a uno los elementos del vector N(). En el SPECTRUM, esta línea graba la configuración de la impresión por impresora.

Línea 1680. Aquí se acaba el bucle y comienza otro dentro de cual se grabarán los nombres de cada campo.

Línea 1690. Se graban los nombres de los campos uno a uno.

Línea 1700. Y se termina el bucle. También se graba el número de fichas de las que consta el fichero.

Línea 1710. Comienza un bucle dentro del cual se grabarán todas las fichas que hay actualmente en el fichero.

Línea 1720. Se graban dichas fichas una a una.

Línea 1730. Aquí termina el bucle encargado de grabar todas la fichas.

Líneas 1740 y **1750.** Comienza un nuevo bucle, esta vez anidado, dentro del cual se grabará la definición de impresión de las fichas.

Línea 1760. Se graban uno a uno los 20 posibles elementos de la matriz N$() que controla el formato de impresión.

Líneas 1770 y **1780.** Aquí se acaba el bucle anidado. En este caso es doble, porque la matriz tiene dimensión dos.

Línea 1790. Se cierra el fichero.

Línea 1800. Y se devuelve el control al programa principal.

Como se ha dicho anteriormente, el grupo de líneas que va desde la **1810** hasta la **1990** se realiza lo mismo que hemos visto anteriormente, pero en cinta.

Hay que decir que los usuarios del SPECTRUM, si no tienen el INTERFACE 1, a la hora de intentar meter el grupo de líneas comprendido entre la **1640** y la **1800,** no van a poder. Esto es debido a que el SPECTRUM sin el INTERFACE 1 no pueden entender los comandos que se tienen que introducir. Según esto, dichos usuarios sólo tendrán que introducir el grupo de líneas que va desde la **1810 hasta la 1990.**

A continuación, y entre las **líneas 2000** y **2160** aparece la parte del programa encargada de gestionar la lectura de fichero. Lógicamente dicha lectura puede hacerse desde un disco o desde una cinta. En este grupo de líneas se crea un nuevo submenú para que el usuario elija su opción y se le pregunta el nombre del fichero a leer. Como hay dos posible opciones de grabación, grabar en disco o en cinta, según lo elegido nos iremos a la **línea 2170** o a la **2360.** Como hemos hecho anteriormente, sólo veremos la primera de las opciones, ya que la segunda es igual y sólo se diferencia en el canal empleada para la grabación.

Línea 2170. Abrimos el fichero que tiene el nombre B$ para leer de él toda las fichas. En el SPECTRUM leemos el vector N().

Línea 2180. Leemos del fichero el valor de N(1). Este nos dice el número de campos que aparecerán a continuación. En el SPECTRUM leemos el vector M$().

Línea 2190. Aquí comenzamos un bucle dentro del cual vamos a ir leyendo toda la configuración del fichero. En el SPECTRUM leemos el vector F$().

Línea 2200. Vamos leyendo uno a uno todos los elementos del vector N(). En el SPECTRUM leemos la matriz N$().

Línea 2210. Aquí se termina el bucle.

Línea 2220. Comenzamos un nuevo bucle dentro del cual vamos a leer los nombres de los campos.

Línea 2230. Y vamos leyendo, uno a uno y según va avanzando el bucle, los nombres de todos los campos.

Línea 2240. Aquí termina el bucle.

Línea 2250. Leemos del fichero el número de fichas que tiene y lo almacenamos en la variable numérica TT.

Línea 2260. Comenzamos un bucle desde uno (1) hasta el número de fichas que tiene el fichero (TT) dentro del cual leeremos dichas fichas.

Línea 2270. Y las vamos leyendo una a una.

Línea 2280. Aquí termina el bucle.

Líneas 2290 y **2300.** Se realiza un bucle anidado dentro del cual se van a leer todos los elementos de la matriz N$(). Esta matriz es la que almacena el formato de impresión de las fichas.

Línea 2310. Leemos los elementos de la matriz N$().

Línea 2320 y **2330.** Aquí termina el bucle anidado.

Línea 2340. Cerramos el fichero.

Línea 2350. Y devolvemos el control al programa principal.

Entre las **líneas 2550** y **2650** vamos a crear un nuevo submenú que le permitirá al usuario: bien imprimir las fichas, bien definir el formato de impresión que desea. Si el usuario elige imprimir las fichas, se transfiere el control del programa a la **línea 2660.** En caso contrario (el usuario desea definir la impresión), nos vamos a la **línea 3390.**

Si el usuario eligió imprimir las fichas, entonces se pregunta si está definido el formato de impresión. En caso negativo, se imprime un mensaje avisándolo y el programa vuelve al menú principal. En caso de que dicho formato sí esté definido, entonces se comprueba que hay fichas en el fichero. Si no las hubiese, el programa informaría de ello y volvería al menú principal.

Cuando, por fin, todas estas pruebas han sido superadas, se le dice al usuario que prepare la impresora y pulse una tecla. En el momento en que lo hace se le dan por pantalla los siguientes datos:

— Los campos se imprimirán.

— El orden de los campos en el papel de la impresora.

— La tabulación donde se imprimirán dichos campos.

— El espacio que habrá entre ficha y ficha. El ordenador nos dice el número de líneas entre ficha y ficha o, en el caso de que esté definido, que habrá un salto de página.

A continuación, y entre las **líneas 2960** y **3190** se realiza la impresión de las fichas. La explicación de estas líneas es la siguiente:

Línea 2960. Se llama a la rutina que está a partir de la **línea 3150** y cuya función es imprimir la cabecera del fichero.

Línea 2970. Se pone la variable CC a cero. Esta variable es auxiliar.

Línea 2980. Comenzamos un bucle entre uno (1) y el número de fichas en el fichero (TT) dentro del cual imprimiremos todas las fichas. También se inicializa la variable CR a uno. Esta variable tiene que valer uno cada vez que el bucle da la vuelta.

Línea 2990. Comenzamos un bucle, que controlará el vector N() dentro del cual imprimiremos todos los campos que el usuario definió. Hacemos que otra variable (CD) valga cero.

Línea 3000. Si resulta que el elemento R,2 del vector de impresión es igual a una cadena vacía ("), entonces es que hemos imprimido todos los campos que teníamos que imprimir de la ficha en curso. En ese caso hacemos que R sea igual a 20 y nos vamos al final del segundo bucle.

Línea 3010. Almacenamos en CC el número del primer campo a imprimir.

Línea 3020. Comenzamos un bucle dentro del cual vamos a buscar dentro de la ficha en curso, en qué posición empieza el campo CC.

Línea 3030. Buscamos el campo CC dentro del registro. Una vez encontrado, almacenamos su posición en la variable Z1.

Línea 3040. Aquí termina el bucle de búsqueda. También en esta línea colocamos la cabeza de la impresora en la columna donde queremos que se imprima el campo que estamos procesando.

Línea 3050. Empieza otro bucle dentro del cual vamos a imprimir dicho campo.

Línea 3060. Como el campo se imprime letra a letra, para controlar el final del mismo se pregunta por el carácter 254. Cuando éste aparece, es que hemos imprimido todos los caracteres. En ese caso hacemos que la variable índice del bucle tenga el máximo valor posible para poder salir del bucle.

Línea 3070. Imprimimos el carácter que corresponda según el valor del bucle.

Línea 3080. Aquí termina el bucle encargado de imprimir el campo en curso.

Línea 3090. Termina el bucle que se encarga de cambiar el campo a imprimir.

Línea 3100. Si el usuario quería cambiar de página cada vez que se imprimiese una ficha, entonces cambiamos de página y volvemos a imprimir la cabecera.

Línea 3110. En caso contrario, se dejan tantas líneas en blanco como se especificaron en la definición.

Línea 3120. Aquí termina el bucle que se va moviendo desde la primera a la última ficha.

Línea 3130. Imprimimos una línea llena de guiones para especificar que se ha terminado la impresión.

Línea 3140. Volvemos al programa principal.

Línea 3150. Aquí empieza la rutina que se encarga de imprimir la cabecera. La cabecera consiste en el nombre de los campos a

imprimir y en una línea de guiones para separar el nombre de los campos de los campos. En esta línea comienza un bucle dentro del cual se imprimirán todos los nombres.

Línea 3160. Si el elemento S,2 de la matriz de impresión N$() es una cadena vacía, entonces es que hemos imprimido todos los nombres de los campos que se van a imprimir. En ese caso hacemos que S sea igual a 20 y nos vamos al final del bucle.

Línea 3170. En caso contrario tabulamos a la posición elegida por el usuario e imprimimos el nombre de dicho campo.

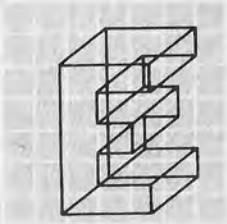
Línea 3180. Aquí se termina el bucle y se imprime una línea de guiones.

Línea 3190. Se devuelve el control a la rutina de impresión.

Entre las **líneas 3230** y **3380** se pone fin al programa, pero, antes de terminarlo, se pregunta al usuario si está seguro de lo que quiere hacer. Esto se ha hecho así como medida de protección ante situaciones que se puedan dar sin que el usuario las desee.

Todavía quedan por explicar unas cuantas rutinas que no se explicaron anteriormente, como son la de visualización de una ficha y la de entrada de fichas nuevas. También hay que darle un repaso a aquéllas que fueron variadas a última hora para aumentar su eficacia y velocidad. Por ello, en el próximo tomo veremos lo poco que nos falta por conocer sobre el programa GESTOR DE FICHEROS.

Reloj de tiempo real



EN el tomo anterior hemos visto cómo pueden realizarse medidas precisas de intervalos de tiempo utilizando el propio reloj del ordenador personal o bien un oscilador adicional y contadores apropiados.

Existen aplicaciones en las cuales es conveniente conocer el día, la hora y hasta el segundo en el que se realiza una medida con mucha precisión. Para estos casos el reloj deducido del que posee el ordenador no suele presentar la precisión y la estabilidad adecuada. Además es normal apagar la máquina cuando no se está utilizando, con lo que se pierde la información almacenada indicando la fecha y la hora. Por estas razones es cada vez más frecuente encontrar en los ordenadores un reloj alimentado por batería que conserva la fecha y la hora aunque se apague la máquina. Este reloj puede venir ya incluido en el equipo inicial o se puede adquirir como una opción.

Se entiende generalmente por reloj de "tiempo real" al que da la indicación de tiempo sin ser afectado por el funcionamiento del sistema y en sincronismo con el tiempo normal del exterior. Es decir, es una reproducción de la hora y la fecha oficial, que no se altera por la ejecución de las operaciones del sistema. La mayoría de las operaciones que realiza el ordenador en comunicación con los periféricos pueden originar la pérdida de una cuenta del reloj interno. Esto es así incluso en el caso de ser la interrupción del reloj la más prioritaria, pues en determinadas circunstancias de error es necesario mantener inhibidas las interrupciones y, por tanto, la cuenta del reloj. Esto no quiere decir que el reloj indicado por el sistema no sea fiable. La hora indicada puede tomarse como indicación aproximada, pero cuando se necesita de verdad la hora de "tiempo real" es necesario utilizar un circuito apropiado para esta función.

Vamos a describir diferentes formas de realizar un reloj de tiempo real accesible desde el ordenador y la programación necesaria para utilizarlo.

Reloj a partir de la señal de red

La tensión de alimentación de los equipos eléctricos domésticos está generada por equipos de muy diferentes tipos pero conectados entre sí, formando una verdadera red. Los generadores disponen de equipos de control muy sofisticados para garantizar unos márgenes de funcionamiento y seguridad adecuados para las aplicaciones domésticas e industriales. Una de las características de mayor importancia es la frecuencia de la señal. Debe ser muy estable, pues pequeñas variaciones originarían alteraciones de los sistemas síncronos que se encuentran conectados.

Por tanto, la señal de red puede tomarse como referencia para la generación de una señal patrón, siempre que nuestro sistema pueda filtrar los ruidos que pueden presentarse por efecto de la conmutación de otros equipos y que tenga algún circuito alternativo para el caso de que falle la red general de alimentación. Es decir, la red es una buena referencia, pero hay que tener previsto que puede faltar, aunque sea por períodos de tiempo pequeños.

El circuito de la figura permite generar una señal de 50 Hz. a partir de la señal del secundario de un transformador. Podría ser el mismo transformador empleado para la obtención de la alimentación de 5 voltios para los circuitos lógicos. Para poder disponer de la señal de red aun cuando no se esté alimentando al sistema, deberemos añadir otro transfor-

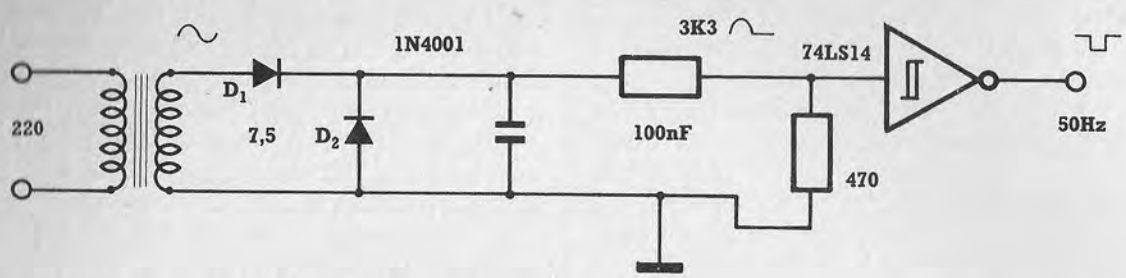


Fig. 1. Generación de señal de 50 Hz. a partir de la red.

mador que esté siempre conectado. Se filtra la señal, para impedir que los ruidos de la red aparezcan en la entrada de los contadores, y después de rectificada se hace cuadrada mediante circuitos con histéresis. El diodo D2 impide que aparezca tensión negativa a la entrada de las puertas, que las dañaría irreversiblemente.

Con la señal de 50 Hz. podemos generar otras frecuencias, mediante contadores divisores. Si utilizamos divisores de décadas, obtendremos los valores de 5, 1 y 0,1 Hz., según se indica en la figura:

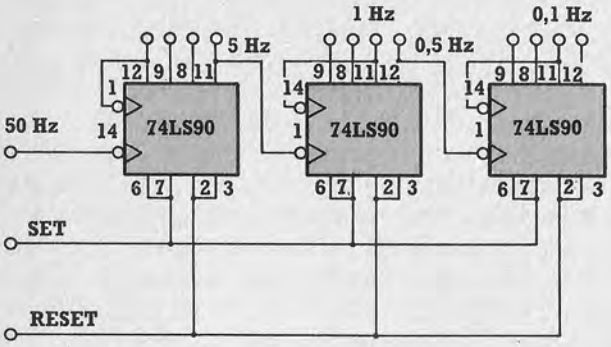


Fig. 2. Contadores divisores.

El circuito 74LS90 es un divisor por 10, separado en un divisor por 2 y un divisor por 5. Puede ponerse a 0 o a 9, según se activen las entradas de Reset o de Set.

Las salidas de las frecuencias de base pueden utilizarse de la misma manera que vimos en el tomo anterior. Para que el sistema pueda funcionar autónomamente, dispondremos de los contadores necesarios para generar un pulso por cada minuto, hora y día. El número de contadores puede parecer elevado, pero se encuentran con facilidad circuitos que incluyen todas las etapas necesarias en un solo circuito integrado. En la figura se muestra uno de ellos que se conectará a la etapa generadora de la señal de referencia.

Reloj basado en circuito con oscilador de cuarzo

Una forma de eliminar la necesidad de la conexión a la red y, por tanto, evitar los microcortes que en ésta pueden producirse es disponer de un oscilador basado en un cristal de cuarzo. Además, son fáciles de encontrar circuitos como el MM5369 que además del oscilador disponen de los divisores para generar una señal de frecuencia igual a la de la red.

El elemento base para la estabilidad es el cristal de cuarzo. La frecuencia de referencia obtenida es de 50 ó 60 Hz., según el tipo. El AA genera 60 Hz. y el EYR genera 50 Hz. El

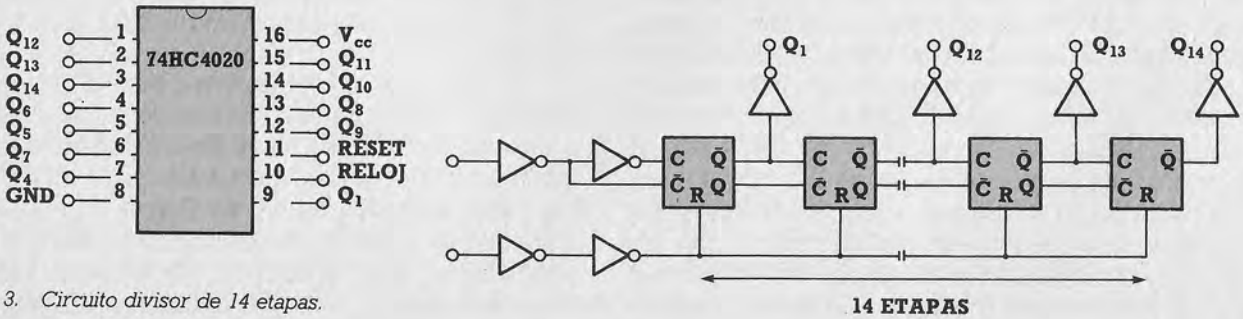


Fig. 3. Circuito divisor de 14 etapas.

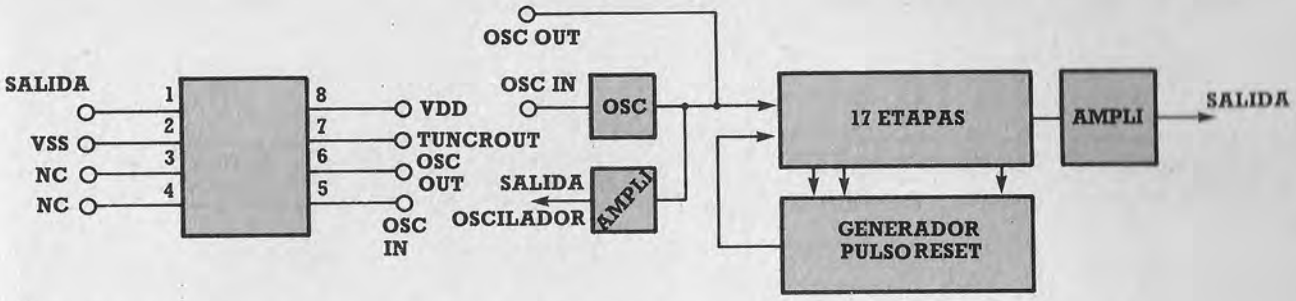


Fig. 4. Reloj de tiempo real utilizando circuito con oscilador de cuarzo.

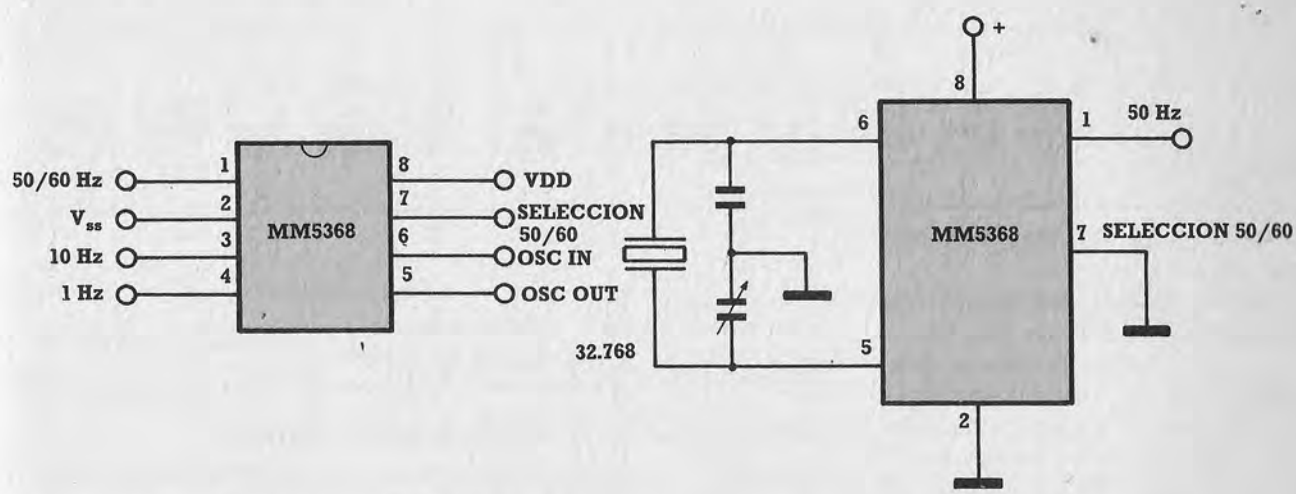


Fig. 5. Reloj de tiempo real con cristal de 32.768 Hz.

cristal es en cualquiera de los casos el de 3,5795 MHz. utilizado en la televisión de norma NTSC. A partir de esta frecuencia generamos las señales de referencia de 1 Hz. y demás con los circuitos de las figuras anteriores.

Otro tipo de circuito integrado permite la generación de la señal de referencia utilizando un cristal de cuarzo como los de los relojes de pulsera.

Este circuito permite obtener también señales de 1 y 10 Hz. Además dispone de una entrada de control para seleccionar los divisores internos que generen la salida fundamental de 50 ó 60 Hz.

El sistema puede en cualquier momento acceder a la hora a través de puertos de entrada conectados a las salidas de los contadores. El programa deberá realizar la conversión de los valores leídos para presentarlos en la forma acostumbrada. Si se utilizan contadores de décadas la conversión es más sencilla. Se hará cada cuatro bits la conversión de BCD a cada uno de los caracteres de la hora. Si los contadores son binarios, será necesario leer el valor total y después hacer una conversión al formato de la hora mediante divisiones sucesivas, tomando el cociente entero como valor de cada cifra y el resto como dividendo para la cifra siguiente.

Reloj con circuitos específicos

Existen circuitos integrados específicos para la generación de la fecha y la hora y que pueden conectarse directamente al bus de un ordenador personal. Todos ellos disponen de los circuitos de direccionamiento para acceder a cada una de las posiciones internas, donde se almacenan la fecha y la hora mientras haya alimentación. Los circuitos están realizados en tecnología CMOS, por lo que presentan muy bajo consumo, lo que les hace fácilmente alimentables con baterías de níquel-cadmio o simplemente con pilas de litio, desechables al cabo de 2 ó 3 años. Ofrecen algunas funciones adicionales como la generación de alarma, la conexión a circuitos de presentación de la hora o la generación de interrupción progra-

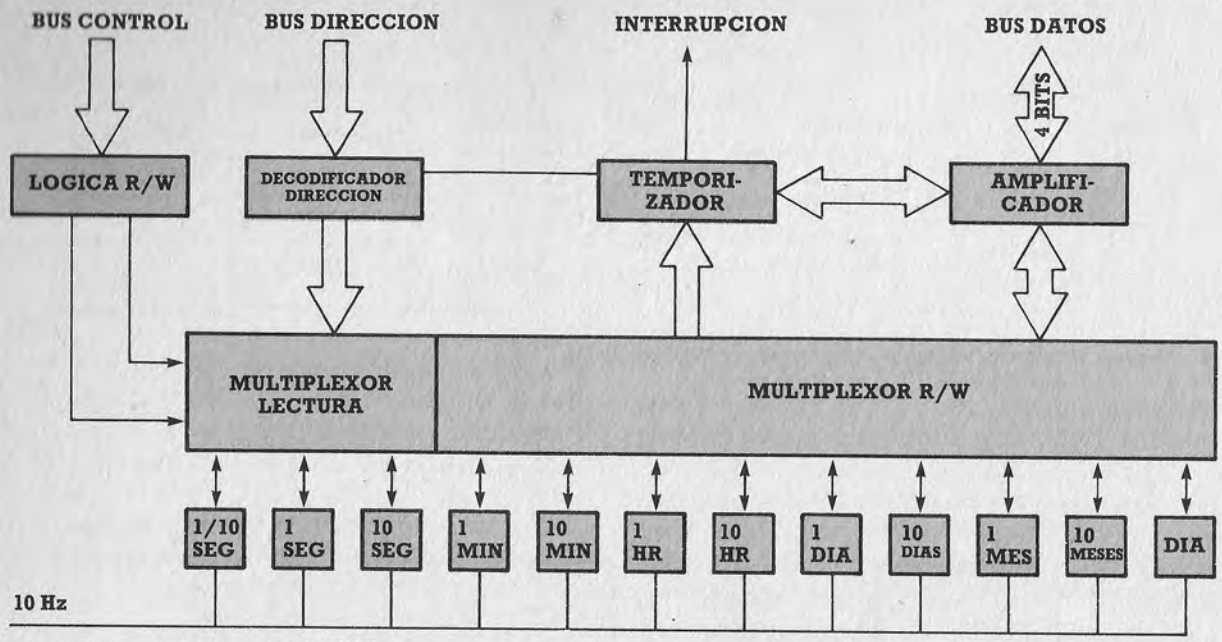


Fig. 6. Diagrama de bloques del MM58174A.

mable para sincronizar el reloj realizado por programa con el de tiempo real.

El consumo es realmente bajo, 12 miliamperios en modo normal y unos 20 microamperios en modo reposo. La tensión de alimentación mínima es de 2,2 voltios. Tienen una entrada de indicación de modo reposo, que desactiva la circuitería interior que debe ponerse en bajo consumo, pasando a circuito abierto los amplificadores de tres estados de comunicación con el bus de datos.

Circuito MM58174A

Es un circuito muy simple en formato de 16 patillas. Es programable y puede generar interrupciones programadas cada medio segundo y 5 y 60 segundos. Utiliza un cristal de cuarzo de 32.768 Hz. Dispone de funcionamiento en bajo consumo. La conexión al bus se hace con 4 bits, siendo cada registro codificado en BCD. Cada registro se accede como una dirección de memoria o de puerto de entrada-salida.

Direcciones y funciones del MM58174A

Según la asignación de los bits de dirección se accede a los diferentes registros internos del circuito, según la siguiente tabla:

Dirección AD3	AD2	AD1	AD0	Registro / Función
0	0	0	0	Prueba
0	0	0	1	Décimas de segundo
0	0	1	0	Segundos
0	0	1	1	Decenas de segundos
0	1	0	0	Minutos
0	1	0	1	Decenas de minutos
0	1	1	0	Horas
0	1	1	1	Decenas de horas
1	0	0	0	Días
1	0	0	1	Decenas de días
1	0	1	0	Día de la semana
1	0	1	1	Mes
1	1	0	0	Decenas de mes
1	1	0	1	Año
1	1	1	0	Arranque / Parada
1	1	1	1	Interrupción y estado

Los registros por encima del 3 pueden ser escritos, para definir la hora y día iniciales.

Circuito MM58167A

Es un circuito algo más complejo, en formato de 24 patillas y que incluye funciones adicionales como interrupción a hora programable.

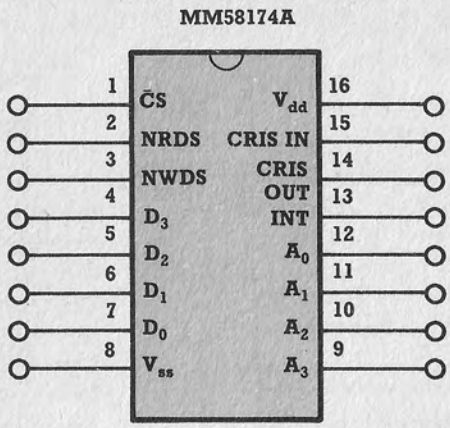


Fig. 7. Patillas del MM58174A.

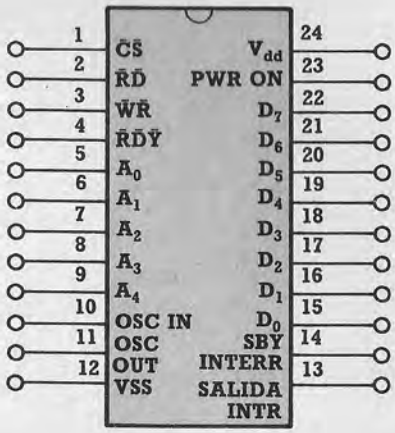
Fig. 8. Diagrama de bloques de MM58167A RTC.

Direcciones y funciones del MM58167A

Este circuito presenta dos juegos de direcciones: uno para el reloj propiamente dicho y otro para la programación de la alarma, según la siguiente tabla:

Dirección AD4	AD3	AD2	AD1	AD0	Registro / Función
0	0	0	0	0	Diezmilésimas de segundo
0	0	0	0	1	Centésimas y décimas de segundo
0	0	0	1	0	Segundos
0	0	0	1	1	Minutos
0	0	1	0	0	Horas
0	0	1	0	1	Día de la semana
0	0	1	1	0	Día del mes
0	0	1	1	1	Mes
0	1	0	0	0	RAM Diezmilésimas de segundo
0	1	0	0	1	RAM Centésimas y décimas de segundo
0	1	0	1	0	RAM Segundos
0	1	0	1	1	RAM Minutos
0	1	1	0	0	RAM Horas
0	1	1	0	1	RAM Día de la semana
0	1	1	1	0	RAM Día del mes
0	1	1	1	1	RAM Mes
1	0	0	0	0	Registro de estado de Interrupción
1	0	0	0	1	Registro de control de Interrupción
1	0	0	1	0	Borrado de contadores
1	0	0	1	1	Borrado de RAM
1	0	1	0	0	Bit de estado
1	0	1	0	1	Comando de arranque
1	0	1	1	0	-Interrupción en espera
1	1	1	1	1	Modo de prueba

Las posiciones de contador o de la RAM de alarma pueden leerse y escribirse según se active la señal de Read o Write del circuito. Si el sistema en el que se instala es rápido, puede usarse la señal de Ready para prolongar el ciclo de lectura o escritura.



El MM56167A se conecta al sistema como un dispositivo de entrada-salida o como memoria. La conexión al bus se hace a 8 hilos. El cristal de cuarzo utilizado es de 32.768 Hz. La alimentación puede bajar hasta 2,2 voltios. Para que a esta tensión pueda mantener la información es necesario que se active la entrada de Pérdida de alimentación (-Pwr Dn).

Ejemplo de aplicación

Veamos la forma de realizar un circuito práctico para dotar a nuestro ordenador personal de un reloj de tiempo real. La conexión podemos hacerla a través de la tarjeta de ampliación de puertos de entrada salida o directamente al bus de datos, si utilizamos las señales de direccionamiento que se indican. Se propone utilizar uno de los relojes más comunes que además utiliza un cristal de cuarzo de los empleados en los relojes de pulsera, por lo que es fácil de encontrar.

Como explicaciones de interés es conveniente mencionar que al disponer internamente de varias posiciones de memoria, es necesario decodificar de forma diferente a como lo hemos hecho para la conexión a través de puertos. Hay que decodificar solamente los bits altos de dirección, hasta A5. Para las máquinas que disponen de poco espacio de direcciones puede montarse una pequeña variación del circuito, que contenga un contador para direccionar cada una de las posiciones. Para leer o actualizar una posición deberemos dar tantos pulsos como correspondan a la dirección deseada, después de haber realizado

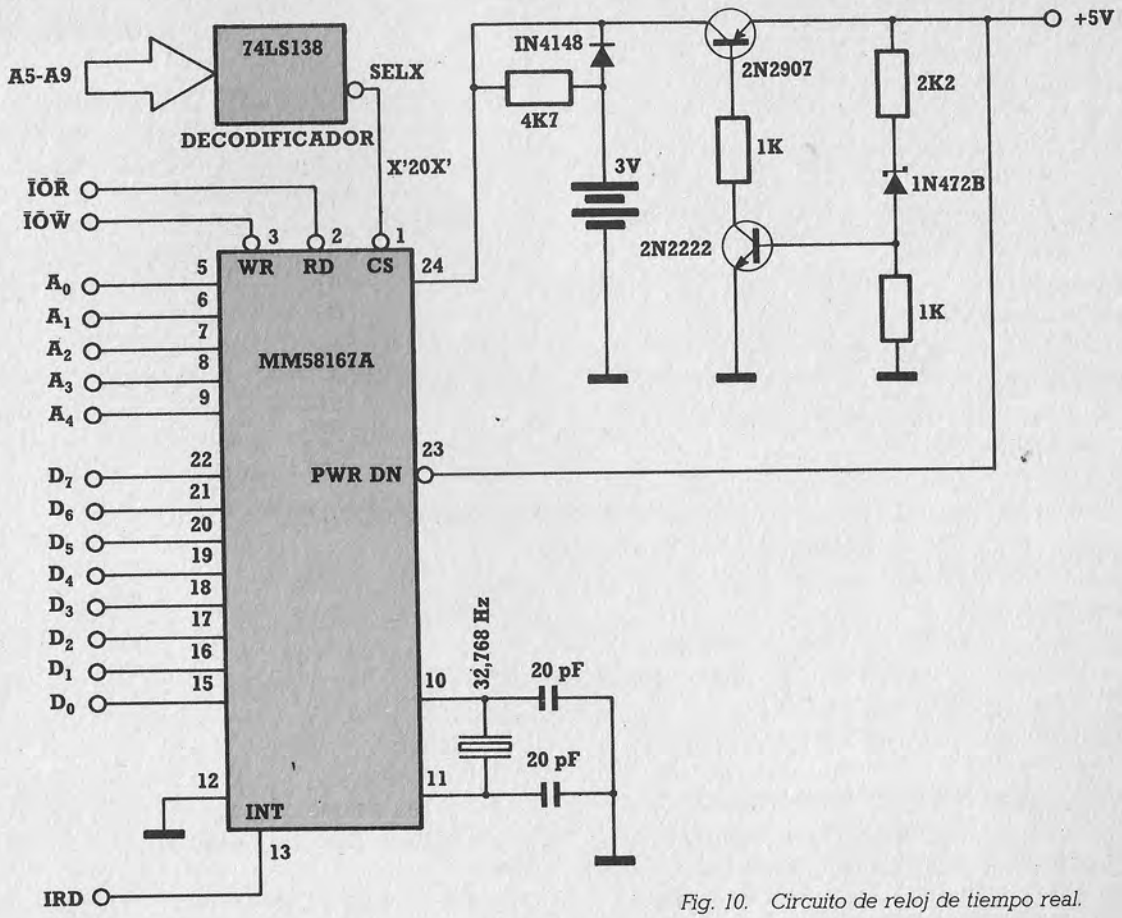


Fig. 10. Circuito de reloj de tiempo real.

un borrado previo. También podría obtenerse la dirección mediante un registro que se direcciona con otra dirección de puerto y se carga, asimismo, con la dirección de la posición a leer.

La alimentación del circuito se hace a través de un regulador, que en condiciones normales de alimentación se conecta directamente a Vcc. Al desaparecer la alimentación general, el diodo empezará a conducir y, por tanto, a alimentar al circuito. Es necesaria la circuitería indicada para garantizar que en el arranque no se borra el contenido de los registros internos.

Programación y lectura del reloj de tiempo real

El programa de ejemplo sirve para inicializar el circuito a la fecha y hora deseada y extraerla posteriormente. Para una aplicación real deberán tomarse las partes del programa que se deseen.

```
10 REM PROGRAMA DE UTILIZACION DEL RELOJ DE TIEMPO REAL
20 REM SE DIRECCIONA MEDIANTE PUERTOS DE E/S
30 REM
40 OPTION BASE 0
50 DEFINT A-Z
60 DIM TMES$(12)
70 DATA "ENERO","FEBRERO","MARZO","ABRIL","MAYO","JUNIO"
80 DATA "JULIO","AGOSTO","SEPTIEMBRE","OCTUBRE","NOVIEMBRE",
   "DICIEMBRE"
90 FOR I=1 TO 12: READ TMES$(I): NEXT I
```

```
100 REM DESCRIPCION DE LAS DIRECCIONES DE PUERTO
110 DIRBASE=&H200: REM DEFINIR LA APROPIADA A CADA MAQUINA
120 CLS
130 PRINT " PROGRAMA DE RELOJ DE TIEMPO REAL"
140 PRINT
150 PRINT " SELECCIONE OPCION:"
160 PRINT
170 PRINT " 1.-PONER FECHA Y HORA"
190 PRINT
200 PRINT " 2.-LEER FECHA Y HORA"
210 PRINT
220 PRINT " 3.-TERMINAR"
260 PRINT
270 INPUT "OPCION=";OPCION
280 IF OPCION<1 OR OPCION>3 THEN BEEP: GOTO 120
290 ON OPCION GOSUB 1000,2000,3000
295 IF OPCION=3 THEN END
300 GOTO 120
1000 REM PONER FECHA Y HORA
1010 PRINT "INDICAR FECHA, NUMERICAMENTE"
1030 INPUT "MES=";MES
1040 INPUT "DIA=";DIA
1050 PRINT
1060 INPUT "HORA";HORA
1070 INPUT "MINUTO";MINUTO
1080 PRINT "LOS VALORES A GRABAR SON:"
1090 PRINT TMES$(MES);" ";DIA;" ";HORA;" ";MINUTO
1100 REM ACTUALIZA EL RELOJ EXTERNO
1110 OUT DIRBASE+7,MES
1120 OUT DIRBASE+6,DIA
1130 OUT DIRBASE+4,HORA
1140 OUT DIRBASE+3,MINUTO
1150 OUT DIRBASE+2,0
1160 OUT DIRBASE+1,0
1170 OUT DIRBASE,0
1180 OUT DIRBASE+21,0: REM COMANDO GO
1200 RETURN
2000 REM LEE LA FECHA Y HORA
2110 MES=INP(DIRBASE+7)
2120 DIA=INP(DIRBASE+6)
2130 HORA=INP(DIRBASE+4)
2140 MINUTO=INP(DIRBASE+3)
2150 SEGUNDO=INP(DIRBASE+2)
2160 PRINT "FECHA=";DIA;" DE ";TMES$(MES)
2170 PRINT "HORA=";HORA;":";MINUTO;":";SEGUNDO
2200 RETURN
3000 PRINT "FIN DEL PROGRAMA"
3010 RETURN
```

Si se configura el circuito como memoria, deberán sustituirse las sentencias INP por PEEK y las OUT por POKE, para realizar la misma función.

Conclusión

Se han presentado varios circuitos que permiten mantener la hora en tiempo real, para utilizarla según convenga en el ordenador. Existen otros muchos circuitos que pueden emplearse para mantener la fecha y la hora en el tiempo real y de fácil conexión al ordenador como son:

— MM58274, de National Semiconductors, versión avanzada del MM58174 descrito al principio.

— MC146818, de Motorola, que incluye además de la circuitería de reloj 64 octetos de RAM CMOS, que sirven para guardar, por ejemplo, datos de configuración de la máquina. Es el actualmente empleado en el IBM-AT.

— ICM7170, de Intersil, que tiene incluso calendario con corrección para años bisiestos y puede emplearse con cuatro diferentes cristales de cuarzo.

La adaptación al ordenador de cualquiera de ellos se hace sin dificultad, dependiendo principalmente de su coste el emplear uno u otro.

NATURALEZA Y TECNOLOGIA QUIMICA

Configuración electrónica

SABEMOS que la materia es divisible y que ya los griegos utilizaron la palabra átomo como límite de divisibilidad. Es cierto que luego han surgido nuevos investigadores que nos han dicho que también la fuerza o un tipo de energía forma parte de la materia. Pero de lo que a continuación vamos a tratar va a ser de los elementos químicos, que si bien no son el origen de toda la materia, sí son aquella clase de materia que no puede descomponerse en otra más sencilla, mediante unos procesos físicos y químicos sencillos (por ejemplo, quedan excluidos los procesos nucleares).

En la actualidad se conocen más de un centenar de elementos, aproximadamente 105.

Sabemos que todos los elementos químicos no son iguales, sino que son diferentes entre sí; entre estas diferencias podemos señalar el número atómico, simbolizado por Z, que es una característica de los átomos de un mismo elemento químico y que representa el número de protones que hay en el núcleo y el número de electrones que hay en la envoltura cuando el átomo está en estado neutro.

La CONFIGURACION ELECTRONICA es una característica más explícita de los átomos de un mismo elemento químico, pues no se limita al recuento de sus electrones, sino que, además, indica otras propiedades, como, por ejemplo, la distribución y la energía de los electrones en la envoltura atómica; los electrones se mueven preferentemente en las regiones del espacio más próximas al núcleo.

```
10 REM ****************************************
20 REM ** ESTRUCTURA ELECTRONICA DE LOS ATOMOS **
30 REM **          SE SUPONE REGULARIDAD        **
40 REM ****************************************
50 REM
60 REM ****************************************
70 REM ** VALIDO PARA AMSTRAD,IBM,MSX,SPECTRUM **
80 REM ****************************************
90 REM
100 DIM H$(19):DIM E(19)
110 FOR N=1 TO 19
120 READ H$(N):NEXT N
130 FOR N=1 TO 19:READ E(N):NEXT N
140 CLS
150 PRINT "*****************E S T R U C T U R A  E L E C T R O N I C A*********
***********"
160 PRINT :PRINT :PRINT :PRINT
170 INPUT "NUMERO ATOMICO:";A
180 PRINT :PRINT :PRINT " LA ESTRUCTURA ELECTRONICA DEL ELEMENTO CUYO NUMERO ATO
MICO ES ";A;" ES:":PRINT:PRINT
190 IF A<1 OR A>118 OR A<>INT(A)THEN GOTO 140
200 LET X=0
210 FOR N=1 TO 19
220 PRINT H$(N);
230 FOR M=1 TO E(N)
240 LET X=X+1:IF X=A THEN GOTO 280
```

```
250 NEXT M
260 PRINT E(N);" ";
270 NEXT N
280 PRINT M;
290 PRINT :PRINT :PRINT :PRINT
300 PRINT "PARA SALIR PULSE S"
310 LET Z$=INKEY$:IF Z$=""THEN GOTO 310
320 IF Z$="S" OR Z$="s" THEN END
330 GOTO 140
340 DATA 1s,2s,2p,3s,3p,4s,3d,4p,5s,4d,5p,6s,4f,5d,6p,7s,5f,6d,7p
350 DATA 2,2,6,2,6,2,10,6,2,10,6,2,14,10,6,2,14,10,6
```

Modificaciones para otros equipos

El programa es válido para el IBM-PC, AMSTRAD y MSX.

COMMODORE

50 PRINT CHR$(147)

SPECTRUM

10 DIM H$ (19,2): DIM E (19)
210 IF Z$="S" OR Z$="s" THEN GO TO 9999

MATEMATICAS

Progresiones

Es famosa la anécdota que se cuenta en todos los libros del inventor del ajedrez, y que dice:

Era Scheran un príncipe indio que conocía todos los juegos de su país y pasaba las tardes aburrido en palacio. Ofreció un premio a aquel que le trajera un juego nuevo que fuera de su agrado. Y fue Sessa, el inventor del ajedrez, el que le presentó su juego al príncipe, y como fue de su agrado ganó el premio ofrecido.

El príncipe Scheran le ofreció en recompensa lo que quisiera, cualquier cosa que pidiera le sería concedida. Y Sessa, que era pobre, no pidió ni tierras ni tesoros, sino que se conformó diciéndole que quería trigo; y la cantidad fue lo que más extrañó al príncipe, ya que pidió: un grano de trigo por el primer cuadrado del tablero de ajedrez, dos granos de trigo por el segundo, cuatro por el tercero, y así sucesivamente hasta llegar a la potencia de 64.

Y se cuenta que el príncipe Scheran no encontró trigo suficiente para recompensar a Sessa en todo su reino.

Si nos fijamos bien, vemos que se trata simplemente de una suma de los 64 primeros términos de una progresión geométrica de razón 2 y cuyo primer término es 1.

Pues bien, con el programa que a continuación presentamos podrás calcular casos similares a éste y muchas operaciones más, relacionadas con las progresiones aritméticas y geométricas y que pueden ser muy útiles en algunos casos.

```
10 REM ****************PROGRESIONES********************
20 CLS
30 PRINT "***************************** M E N U ******************************
**********"
40 PRINT :PRINT :PRINT :PRINT
50 PRINT "              1 PROGRESIONES ARITMETICAS"
60 PRINT :PRINT
70 PRINT "              2 PROGRESIONES GEOMETRICAS"
80 PRINT :PRINT
90 PRINT "              3 FIN DEL PROGRAMA"
100 PRINT:PRINT:INPUT "OPCION ELEGIDA:";A
110 IF A=1 THEN GOTO 150
120 IF A=2 THEN GOTO 600
130 IF A=3 THEN END
140 GOTO 20
150 REM ****************PROGRESIONES ARITMETICAS*******************
160 CLS
170 PRINT "******************* P R O G R E S I O N E S  A R I T M E T I C A S**
***********"
180 PRINT :PRINT
190 INPUT "PRIMER TERMINO:";P
200 INPUT "DIFERENCIA ENTRE TERMINOS:";D
```

```
210 GOSUB 500
220 PRINT "                    1 SUMA DE N TERMINOS"
230 PRINT :PRINT
240 PRINT "                    2 CALCULO DEL TERMINO N"
250 PRINT :PRINT
260 PRINT "PULSA OPCION (S PARA SALIR)"
270 LET A$=INKEY$:IF A$="" THEN GOTO 270
280 IF A$="1" THEN GOTO 310
290 IF A$="2" THEN GOTO 420
300 IF A$="S" OR A$="s" THEN GOTO 20
310 GOSUB 500
320 INPUT "SUMA DESDE EL TERMINO NUMERO";X
330 IF X<=0 THEN GOTO 320
340 INPUT "HASTA EL TERMINO NUMERO";Y
350 IF X>Y THEN GOTO 340
360 LET J=P+D*(X-1):LET K=P+D*(Y-1)
370 PRINT "SUMA DESDE EL TERMINO ";X ;" HASTA EL TERMINO ";Y;"=";(J+K)*(Y-X+1)/2
380 PRINT :PRINT:PRINT "OTRA SUMA?"
390 LET A$=INKEY$:IF A$="" THEN GOTO 390
400 IF A$="S" OR A$="s" THEN GOTO 310
410 GOTO 210
420 GOSUB 500
430 INPUT "NUMERO DEL TERMINO A CALCULAR";B
440 IF B<=0 THEN GOTO 430
450 PRINT :PRINT:PRINT" a";B;"=";P+D*(B-1)
460 PRINT :PRINT:PRINT "OTRO TERMINO?"
470 LET A$=INKEY$:IF A$="" THEN GOTO 470
480 IF A$="S" OR A$="s" THEN GOTO 420
490 GOTO 210
500 CLS
510 PRINT "          **** P R O G R E S I O N   A R I T M E T I C A ****"
515 PRINT:PRINT:PRINT:PRINT
520 PRINT "                                a1=";P
530 PRINT "                                 D=";D
540 PRINT :PRINT
550 RETURN
600 REM ****************PROGRESIONES GEOMETRICAS*******************
610 CLS
620 PRINT "******************* P R O G R E S I O N E S  G E O M E T R I C A S**
***********"
630 PRINT :PRINT
640 INPUT "PRIMER TERMINO:";P
650 INPUT "RAZON DE LA PROGRESION:";R
660 GOSUB 2000
670 PRINT "                    1 SUMA DE N TERMINOS"
680 PRINT :PRINT
690 PRINT "                    2 PRODUCTO DE N TERMINOS"
700 PRINT :PRINT
710 PRINT "                    3 CALCULO DEL TERMINO N"
720 PRINT :PRINT
730 PRINT "PULSA OPCION (S PARA SALIR)"
740 LET A$=INKEY$:IF A$="" THEN GOTO 740
750 IF A$="1" THEN GOTO 800
760 IF A$="2" THEN GOTO 900
770 IF A$="3" THEN GOTO 1000
780 IF A$="S" OR A$="s" THEN GOTO 20
790 GOTO 660
800 GOSUB 2000
810 INPUT "SUMA DESDE EL TERMINO NUMERO";X
820 IF X<=0 THEN GOTO 810
830 INPUT "HASTA EL TERMINO NUMERO";Y
840 IF X>Y THEN GOTO 830
850 LET J=P*(R^(X-1)):LET K=P*(R^(Y-1))
860 PRINT "SUMA DESDE EL TERMINO ";X ;" HASTA EL TERMINO ";Y;"=";(K*R-J)/(R-1)
870 PRINT :PRINT:PRINT "OTRA SUMA?"
880 LET A$=INKEY$:IF A$="" THEN GOTO 880
890 IF A$="S" OR A$="s" THEN GOTO 800
895 GOTO 660
900 GOSUB 2000
910 INPUT "PRODUCTO DESDE EL TERMINO NUMERO";X
920 IF X<=0 THEN GOTO 910
930 INPUT "HASTA EL TERMINO NUMERO";Y
940 IF X>Y THEN GOTO 930
950 LET J=P*(R^(X-1)):LET K=P*(R^(Y-1))
960 PRINT "PRODUCTO DESDE EL TERMINO ";X ;" HASTA EL TERMINO ";Y;"=";(K*J)^((Y-X
+1)/2)
970 PRINT :PRINT:PRINT "OTRO PRODUCTO?"
980 LET A$=INKEY$:IF A$="" THEN GOTO 980
990 IF A$="S" OR A$="s" THEN GOTO 900
995 GOTO 660
```

```
1000 GOSUB 2000
1010 INPUT "NUMERO DEL TERMINO A CALCULAR";B
1020 IF B<=0 THEN GOTO 430
1030 PRINT :PRINT:PRINT" a";B;"=";P*(R^(B-1))
1040 PRINT :PRINT:PRINT "OTRO TERMINO?"
1050 LET A$=INKEY$:IF A$="" THEN GOTO 1050
1060 IF A$="S" OR A$="s" THEN GOTO 1000
1070 GOTO 660
2000 CLS
2010 PRINT "          **** P R O G R E S I O N   G E O M E T R I C A ****"
2020 PRINT:PRINT:PRINT:PRINT
2030 PRINT "                                a1=";P
2040 PRINT "                                 R=";R
2050 PRINT :PRINT
2060 RETURN
```

Modificaciones para otros equipos

El programa es válido para el IBM-PC, AMSTRAD y MSX.

COMMODORE

Cambiar los CLS por PRINT CHR$(147)

SPECTRUM

130 IF A=3 THEN GO TO 9999

SOCIEDAD

Horarios en el mundo

Todos sabemos que la Tierra, en su movimiento de traslación alrededor del Sol, pasa por distintas posiciones con relación a éste; pero esas posiciones ocurren en grandes períodos de tiempo y quizá no lo vemos claramente. De lo que sí nos damos cuenta perfectamente es de cuándo amanece y de cuándo se pone el Sol.

Pues bien, es lógico que esto, que es tan claro a simple vista, tenga cierta complicación en un planeta como el nuestro; en efecto, lo que ocurre es que no en todos los países amanece y anochece al mismo tiempo, sino que cada país, dependiendo de su situación geográfica, recibe los rayos del Sol en distintos momentos que los demás países.

Esto crea unas diferencias horarias tan grandes en algunos casos, que cuando en un país está saliendo el Sol es tiempo del ocaso en otro.

Figúrate que tuvieras que hacer un viaje o que simplemente por curiosidad te interesara conocer la hora de otro país. ¿Cómo podrías saberlo? En este programa que a continuación te presentamos te ofrecemos la manera, si no más rápida, sí más sencilla de conocer el horario de otros países que pueden estar muy lejos del nuestro.

Para la construcción de este programa se han empleado tres matrices: la primera para almacenar los nombres de los países de los que el ordenador conoce la hora. En la segunda y tercera las horas y minutos, respectivamente, de la diferencia horaria con respecto a la Península.

Si el ordenador no conoce la hora del lugar solicitado, pregunta, hasta en cinco ocasiones, un país próximo al introducido, para calcular una aproximación horaria. Si todos estos países también le fueran desconocidos, el programa responde ignorando el horario de ese lugar.

```
10 REM ***************************************************
20 REM ***            HORARIOS EN EL MUNDO             ***
30 REM ***************************************************
40 DIM P$(65):DIM Q(65):DIM R(65)
50 FOR I=1 TO 65
60 READ P$(I)
70 NEXT I
80 FOR I=1 TO 65
90 READ Q(I)
100 NEXT I
110 FOR I=1 TO 65
120 READ R(I)
130 NEXT I
```

```
140 CLS
150 PRINT "********************** I N S T R U C C I O N E S ******************
***********"
160 PRINT:PRINT
170 PRINT "                EN PRIMER LUGAR DEBERAS INTRODUCIR"
180 PRINT "                   LA HORA Y DESPUES LOS MINUTOS"
190 PRINT "                         EN LA PENINSULA"
200 PRINT
210 PRINT"               A CONTINUACION TENDRAS QUE DECIRLE"
220 PRINT"                 A TU ORDENADOR EL PAIS DEL QUE"
230 PRINT "                      QUIERES CONOCER LA HORA"
240 PRINT :PRINT
250 PRINT "         EJEMPLO:"
260 PRINT "                          HORA:17"
270 PRINT "                        MINUTOS:30"
280 PRINT
290 PRINT "                        PAIS   :MARRUECOS"
300 PRINT
310 PRINT "         LAS 17:30 EN LA PENINSULA SERAN LAS 16:30 EN MARRUECOS"
320 PRINT
330 PRINT "                 *PULSA UNA TECLA PARA COMENZAR*"
340 LET A$=INKEY$:IF A$="" THEN GOTO 340
350 CLS:LET A=0
360 PRINT "****************** H O R A R I O S  E N  E L  M U N D O *************
***********"
370 PRINT:PRINT
380 PRINT "                      *HORA EN LA PENINSULA*"
390 PRINT
400 INPUT"                              HORA:";H
410 IF H<0 OR H>=24 THEN GOTO 400
420 INPUT"                            MINUTOS:";M
430 IF M<0 OR M>=60 THEN GOTO 420
440 PRINT
450 INPUT"                              PAIS:";N$
460 REM ********************** BUSQUEDA DEL PAIS **********************
470 LET C=0:LET H1=H:LET M1=M
480 FOR I=1 TO 65
490 IF P$(I)=N$ THEN GOTO 570
500 NEXT I
510 LET C=C+1:IF C=5 THEN GOTO 540
520 PRINT "           **DAME UN PAIS CERCANO A ";N$;"**"
530 INPUT"                              PAIS:";N$:GOTO 480
540 PRINT "           LO SIENTO. NO CONOZCO EL HORARIO DE ESA ZONA"
550 PRINT:PRINT  "        PULSA UNA TECLA PARA CONTINUAR (S PARA SALIR)"
560 GOTO 730
570 LET M=M+R(I)
580 IF M>=60 THEN M=M-60:LET H=H+1:GOTO 600
590 IF M<=0 THEN M=60+M:LET H=H-1
600 LET H=H+Q(I)
610 IF H>=24 THEN H=H-24:LET A=1:GOTO 630
620 IF H<0 THEN H=24+H:LET A=-1
630 PRINT
640 PRINT "         LAS ";H1;":";M1;" EN LA PENINSULA SERAN LAS ";H;":";M;" EN "
;N$
650 IF A=1 THEN PRINT "                 DEL DIA SIGUIENTE"
660 IF A=-1 THEN PRINT "                 DEL DIA ANTERIOR"
670 IF I=12 THEN GOSUB 760
680 IF I=15 THEN GOSUB 780
690 IF I=24 THEN GOSUB 800
700 IF I=60 THEN GOSUB 820
710 IF I=62 THEN GOSUB 840
720 PRINT:PRINT  "        PULSA UNA TECLA PARA CONTINUAR (S PARA SALIR)"
730 LET Z$=INKEY$:IF Z$="" THEN GOTO 730
740 IF Z$="S" OR Z$="s" THEN END
750 GOTO 350
760 PRINT " HORA DEL ESTE.  EN BRASIL(OESTE) SERA UNA HORA MENOS"
770 RETURN
780 PRINT " HORA DE LA ZONA ATLANTICA.,EN  LA COSTA DEL PACIFICO DE CANADA SERAN
          CUATRO HORAS MENOS"
790 RETURN
800 PRINT "                EN CANARIAS SERA UNA HORA MENOS"
810 RETURN
820 PRINT " HORA DE LA CAPITAL.EN LA ZONA DE VLADIVOSTOK SERAN SIETE HORAS MAS"
830 RETURN
840 PRINT " HORA DE LA CAPITAL.  EN LA ZONA CENTRO SERA UNA HORA MENOS"
850 PRINT "             EN LA COSTA DEL PACIFICO SERAN TRES HORAS MENOS"
860 PRINT "             EN LA ZONA DE HAWAI SERAN CINCO HORAS MENOS"
870 RETURN
880 DATA AFGANISTAN,ALEMANIA,ARABIA SAUDITA, ARGELIA,ARGENTINA,AUSTRALIA,AUSTRIA
,BAHRAIN,BELGICA,BERMUDAS
```

```
890 DATA BOLIVIA,BRASIL,BULGARIA,CAMERUN,CANADA,COLOMBIA,CONGO,CUBA
900 DATA CHECOSLOVAQUIA,CHILE,CHINA,DINAMARCA,EGIPTO,ESPAÑA,FILIPINAS,FINLANDIA,
FRANCIA,GRAN BRETAÑA,GRECIA
910 DATA GUATEMALA,HOLANDA,HONDURAS,HUNGRIA,INDIA,IRAK,IRAN,IRLANDA,ISRAEL,ITALI
A
920 DATA JAPON,MEXICO.MOZAMBIQUE,NICARAGUA,NORUEGA,NUEVA ZELANDA,PAKISTAN,PANAMA
,PARAGUAY,PERU
930 DATA POLONIA,PORTUGAL,RUMANIA,SIRIA,SUDAFRICA,SUECIA,SUIZA,TAILANDIA,TURQUIA
,UGANDA
940 DATA U.R.S.S.,URUGUAY,U.S.A.,VENEZUELA,YUGOSLAVIA,ZIMBABWE
950 DATA 3,0,2,-1,-4,9,0,2,0,-5,-5,-4,1,0,-5,-6,0,-6,0,-5,7,0,1,0,7,1
960 DATA 0,-1,1,-7,0,-7,0,4,2,2,-1,1,0,+8,-7,1,-7,0,11,4,-6,-5,-6,0,-1,1,1,1,0,0
,6,1,2,2,-4,-6,-5,0,1
970 DATA 30,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,30,0
,30,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
```

Modificaciones para otros equipos

El programa es válido para el IBM-PC, AMSTRAD y MSX.

COMMODORE

Cambiar todos los CLS por PRINT CHR$(147)

SPECTRUM

40 DIM P$(65,20):DIM Q(65):DIM R(65)
740 IF Z$="S" OR Z$="s" THEN GO TO 9999
Todos los DATA alfanuméricos deberán ser introducidos entre comillas.

PARA LOS MAS JOVENES

Estructuras matemáticas

Existen partes de la Matemática cuyo uso no es muy corriente, y que no tienen la aplicación práctica que pueden tener otras, pero no por ello van a ser menos importantes que las demás. Su uso, aunque indirectamente, puede ser tan necesario como el de cualquier otro dominio de la Matemática.

Hay muchos casos en los que un problema o un ejercicio se simplifica notablemente si conocemos las propiedades que en ese caso actúan; de ese modo podemos evitar, a veces, gran cantidad de cálculos, que hubieran sido innecesarios.

También el conocimiento de las propiedades que cumplen algunos conjuntos con sus leyes operativas permite comparar unos con otros y elaborar teorías discriminatorias, así como prever el comportamiento de dichos conjuntos en situaciones concretas.

En el programa que a continuación presentamos nos basamos en una gran base de datos, de la que vamos a hacer uso, según necesitemos, a través de diversos menús, que irán mostrando en la pantalla la estructura de los distintos conjuntos de números con las leyes operativas que nosotros solicitemos.

```
10 REM **********************
20 REM ** TEORIA DE GRUPOS **
30 REM **********************
40 REM
50 REM ****************************************
60 REM ** VALIDO PARA IBM,MSX,SPECTRUM,AMSTRAD **
70 REM ****************************************
80 REM
90 REM
100 CLS:PRINT "******************************** M E N U ********************
***************"
110 PRINT:PRINT :PRINT
120 PRINT TAB(25);"1 ESTRUCTURA DE GRUPO":PRINT:PRINT
130 PRINT TAB(25);"2 ESTRUCTURA DE SEMIGRUPO":PRINT:PRINT
140 PRINT TAB(25);"3 ESTRUCTURA DE ANILLO":PRINT:PRINT
150 PRINT TAB(25);"4 ESTRUCTURA DE CUERPO":PRINT:PRINT
160 PRINT:INPUT "OPCION ELEGIDA (U PARA SALIR):";A
165 IF A=U THEN END
170 IF A>4 OR A<1 THEN GOTO 100
180 ON A GOTO 200,500,700,900
200 CLS:PRINT"***************** E S T R U C T U R A  D E  G R U P O **********
*************"
210 PRINT:PRINT
```

```
220 PRINT "LOS CONJUNTOS QUE PRESENTAN ESTRUCTURA DE GRUPO SON:"
230 PRINT:PRINT
240 PRINT TAB(15);"A) NUMEROS ENTEROS RESPECTO A LA SUMA"
250 PRINT:PRINT
260 PRINT TAB(15);"B) NUMEROS RACIONALES RESPECTO A LA SUMA"
270 PRINT:PRINT
280 PRINT TAB(15);"C) NUMEROS RACIONALES RESPECTO AL PRODUCTO"
290 PRINT:PRINT
300 PRINT TAB(15);"D) NUMEROS REALES RESPECTO A LA SUMA"
310 PRINT:PRINT
320 PRINT TAB(15);"E) NUMEROS REALES RESPECTO AL PRODUCTO"
325 PRINT:PRINT
330 PRINT "PULSA LA LETRA DEL CONJUNTO QUE QUIERAS VER (M PARA VOLVER AL MENU PR
INCIPAL)"
340 LET A$=INKEY$:IF A$="" THEN GOTO 340
343 IF A$="M" OR A$="m" THEN GOTO 100
345 IF A$<>"A" AND A$<>"B" AND A$<>"C" AND A$<>"D" AND A$<>"E" THEN GOTO 200
350 IF A$="A" THEN H$="ENTEROS":LET K$="ENTERO"
360 IF A$="B" OR A$="C" THEN H$="RACIONALES":LET K$="RACIONAL"
370 IF A$="D" OR A$="E" THEN H$="REALES":LET K$="REAL"
380 IF A$="A" OR A$="B" OR A$="D" THEN J$=" LA SUMA":LET O$="+":GOTO 400
390 LET J$="L PRODUCTO":LET O$="x"
400 GOTO 1000
500 CLS:PRINT"***************** E S T R U C T U R A  D E  S E M I G R U P O ***
**************"
510 PRINT:PRINT
520 PRINT "LOS CONJUNTOS QUE PRESENTAN ESTRUCTURA DE SEMIGRUPO SON:"
530 PRINT:PRINT:PRINT:PRINT
540 PRINT TAB(15);"A) NUMEROS NATURALES RESPECTO A LA SUMA"
550 PRINT:PRINT
560 PRINT TAB(15);"B) NUMEROS NATURALES RESPECTO AL PRODUCTO"
570 PRINT:PRINT
580 PRINT TAB(15);"C) NUMEROS ENTEROS RESPECTO AL PRODUCTO"
590 PRINT:PRINT :PRINT :PRINT
600 PRINT "PULSA LA LETRA DEL CONJUNTO QUE QUIERAS VER (M PARA VOLVER AL MENU PR
INCIPAL)"
610 LET A$=INKEY$:IF A$="" THEN GOTO 610
620 IF A$="M" OR A$="m" THEN GOTO 100
625 IF A$<>"A" AND A$<>"B" AND A$<>"C" THEN GOTO 500
630 IF A$="A" OR A$="B" THEN H$="NATURALES":LET K$="NATURAL":GOTO 650
640 LET H$="ENTEROS":LET K$="ENTERO"
650 IF A$="A" THEN J$=" LA SUMA":LET O$="+":GOTO 670
660 LET J$="L PRODUCTO":LET O$="x"
670 GOTO 2000
700 CLS:PRINT"***************** E S T R U C T U R A  D E  A N I L L O *********
**************"
710 PRINT:PRINT
720 PRINT "EL CONJUNTO QUE PRESENTA ESTRUCTURA DE ANILLO ES:"
730 PRINT:PRINT:PRINT:PRINT
740 PRINT TAB(15);"A) NUMEROS ENTEROS RESPECTO A LA SUMA Y AL PRODUCTO"
745 PRINT:PRINT:PRINT:PRINT:PRINT
750 PRINT "PULSA LA LETRA A PARA VER ESTE CONJUNTO (M PARA VOLVER AL MENU PRINCI
PAL)"
760 LET A$=INKEY$:IF A$="" THEN GOTO 760
770 IF A$="M" OR A$="m" THEN GOTO 100
780 IF A$<>"A" THEN GOTO 700
790 CLS:PRINT "          *ESTRUCTURA DE ANILLO DEL CONJUNTO DE LOS NUMEROS ENTER
OS*":PRINT"          *RESPECTO A LA SUMA Y AL PRODUCTO*"
800 PRINT:PRINT
810 PRINT "   - LA SUMA DE LOS NUMEROS ENTEROS TIENE ESTRUCTURA DE GRUPO CON RES
PECTO A          LA SUMA"
820 PRINT:PRINT
830 PRINT "   - EL PRODUCTO DE LOS NUMEROS ENTEROS TIENE ESTRUCTURA DE SEMIGRUPO
 "
840 PRINT:PRINT
850 PRINT "   - EL ELEMENTO NEUTRO DEL PRODUCTO ES EL 1"
860 PRINT:PRINT
870 PRINT "   - LA SUMA Y EL PRODUCTO TIENEN LA PROPIEDAD CONMUTATIVA:"
880 PRINT "                    a+b=b+a  //   axb=bxa"
885 PRINT:PRINT:PRINT
890 PRINT "         *LOS NUMEROS ENTEROS FORMAN UN ANILLO UNITARIO Y CONMUTATIVO
*"
895 PRINT:PRINT:PRINT"                PULSA UNA TECLA PARA VOLVER AL MENU PRINCI
PAL"
897 LET Z$=INKEY$:IF Z$="" THEN GOTO 897
899 GOTO 100
900 CLS:PRINT"***************** E S T R U C T U R A  D E  C U E R P O ********
**************"
910 PRINT:PRINT
920 PRINT "LOS CONJUNTOS QUE PRESENTAN ESTRUCTURA DE CUERPO SON:"
```

```
930 PRINT:PRINT:PRINT:PRINT
940 PRINT TAB(15);"A) NUMEROS RACIONALES RESPECTO A LA SUMA Y AL PRODUCTO"
950 PRINT:PRINT:PRINT:PRINT
960 PRINT TAB(15);"B) NUMEROS REALES RESPECTO A LA SUMA Y AL PRODUCTO"
970 PRINT:PRINT:PRINT:PRINT:PRINT
975 PRINT "PULSA LA LETRA DEL CONJUNTO QUE QUIERAS VER (M PARA VOLVER AL MENU PR
INCIPAL)"
980 LET A$=INKEY$:IF A$="" THEN GOTO 980
985 IF A$="M" OR A$="m" THEN GOTO 100
987 IF A$<>"A" AND A$<>"B" THEN GOTO 900
990 IF A$="A" THEN GOTO 2400
995 IF A$="B" THEN GOTO 2500
1000 CLS:PRINT "***ESTRUCTURA DE GRUPO DE LOS NUMEROS ";H$;" RESPECTO A";J$;"***
"
1010 PRINT:PRINT
1020 PRINT TAB(25);"1) LEY DE COMPOSICION INTERNA"
1030 PRINT:PRINT
1040 PRINT TAB(25);"2) ASOCIATIVA"
1050 PRINT:PRINT
1060 PRINT TAB(25);"3) ELEMENTO NEUTRO"
1070 PRINT:PRINT
1080 PRINT TAB(25);"4) ELEMENTO SIMETRICO U OPUESTO"
1090 PRINT:PRINT
1100 INPUT "¿QUE PROPIEDAD QUIERES VER?(0 PARA SALIR)";P
1110 IF P<0 OR P>4 THEN GOTO 1100
1120 IF P=0 THEN GOTO 200
1130 ON P GOSUB 1300,1400,1500,1600
1140 GOTO 1000
1300 CLS:PRINT "*************************LEY DE COMPOSICION INTERNA***********
****************"
1310 PRINT:PRINT:PRINT:PRINT
1320 PRINT "                    SI a Y b SON DOS NUMEROS ";H$
1330 PRINT:PRINT
1340 PRINT "                  ENTONCES a";O$;"b ES OTRO NUMERO ";K$
1350 GOTO 3000
1400 CLS:PRINT "******************************PROPIEDAD ASOCIATIVA**************
****************"
1410 PRINT:PRINT:PRINT:PRINT
1420 PRINT "                     a,b,c SON TRES NUMEROS ";H$
1430 PRINT:PRINT
1440 PRINT "                  (a";O$;"b)";O$;"c=a";O$;"(b";O$;"c)"
1450 GOTO 3000
1500 CLS:PRINT "*******************************ELEMENTO NEUTRO****************
****************"
1510 PRINT:PRINT:PRINT:PRINT
1520 IF A$="C" OR A$="E" THEN Y =1:GOTO 1540
1530 LET Y=0
1540 PRINT "          EL ELEMENTO NEUTRO DE LOS NUMEROS ";H$;" ES EL ";Y
1550 PRINT:PRINT
1560 PRINT "                SIENDO a UN NUMERO ";K$
1570 PRINT "                     SE VERIFICA QUE"
1580 PRINT "                       a";O$;Y;"=a"
1590 GOTO 3000
1600 CLS:PRINT "******************************ELEMENTO SIMETRICO***************
**************"
1610 PRINT:PRINT:PRINT:PRINT
1620 IF A$="A" OR A$="B" OR A$="D" THEN LET X$="-a":GOTO 1640
1630 LET X$="1/a"
1640 PRINT "        SIENDO a UN NUMERO ";K$
1650 PRINT "          EL ELEMENTO SIMETRICO DE a ES ";X$
1660 PRINT:PRINT
1670 PRINT "                     YA QUE SE VERIFICA:     "
1680 PRINT "                       a";O$;X$;"=ELEMENTO NEUTRO"
1690 GOTO 3000
2000 CLS:PRINT "***ESTRUCTURA DE SEMIGRUPO DE LOS NUMEROS ";H$;" RESPECTO A";J$;
"***"
2010 PRINT:PRINT
2020 PRINT TAB(25);"1) LEY DE COMPOSICION INTERNA"
2030 PRINT:PRINT
2040 PRINT TAB(25);"2) ASOCIATIVA"
2050 PRINT:PRINT
2060 INPUT "¿QUE PROPIEDAD QUIERES VER?(0 PARA SALIR)";P
2070 IF P<0 OR P>2 THEN GOTO 2060
2080 IF P=0 THEN GOTO 500
2090 ON P GOSUB 2200,2300
2100 GOTO 2000
2200 CLS:PRINT "*************************LEY DE COMPOSICION INTERNA***********
****************"
2210 PRINT:PRINT:PRINT:PRINT
2220 PRINT "                    SI a Y b SON DOS NUMEROS ";H$
```

```
2230 PRINT:PRINT
2240 PRINT "                    ENTONCES a";O$;"b ES OTRO NUMERO ";K$
2250 GOTO 3000
2300 CLS:PRINT "*****************************PROPIEDAD ASOCIATIVA*************
****************"
2310 PRINT:PRINT:PRINT:PRINT
2320 PRINT "                        a,b,c SON TRES NUMEROS ";H$
2330 PRINT:PRINT
2340 PRINT "                    (a";O$;"b)";O$;"c=a";O$;"(b";O$;"c)"
2350 GOTO 3000
2400 CLS:PRINT "            *ESTRUCTURA DE CUERPO DEL CONJUNTO DE LOS NUMEROS RACI
ONALES*":PRINT"                 *RESPECTO A LA SUMA Y AL PRODUCTO*"
2410 PRINT:PRINT
2420 PRINT "   - LA SUMA DE LOS NUMEROS RACIONALES TIENE ESTRUCTURA DE GRUPO "
2430 PRINT:PRINT
2440 PRINT "   - EL PRODUCTO DE LOS NUMEROS RACIONALES TIENE ESTRUCTURA DE GRUPO
 "
2450 PRINT:PRINT
2460 PRINT "   - LA SUMA Y EL PRODUCTO TIENEN LA PROPIEDAD CONMUTATIVA:"
2470 PRINT "                        a+b=b+a  //   axb=bxa"
2480 PRINT:PRINT:PRINT
2490 PRINT "          &LOS NUMEROS RACIONALES FORMAN UN CUERPO ABELIANO O CONMUTA
TIVO&"
2495 PRINT:PRINT:PRINT"                 PULSA UNA TECLA PARA VOLVER AL MENU PRINC
IPAL"
2497 LET Z$=INKEY$:IF Z$="" THEN GOTO 897
2499 GOTO 100
2500 CLS:PRINT "            *ESTRUCTURA DE CUERPO DEL CONJUNTO DE LOS NUMEROS REAL
ES*":PRINT"                 *RESPECTO A LA SUMA Y AL PRODUCTO*"
2510 PRINT:PRINT
2520 PRINT "   - LA SUMA DE LOS NUMEROS REALES TIENE ESTRUCTURA DE GRUPO "
2530 PRINT:PRINT
2540 PRINT "   - EL PRODUCTO DE LOS NUMEROS REALES TIENE ESTRUCTURA DE GRUPO "
2550 PRINT:PRINT
2560 PRINT "   - LA SUMA Y EL PRODUCTO TIENEN LA PROPIEDAD CONMUTATIVA:"
2570 PRINT "                        a+b=b+a  //   axb=bxa"
2580 PRINT:PRINT:PRINT
2590 PRINT "          &LOS NUMEROS REALES FORMAN UN CUERPO ABELIANO O CONMUTATIVO
&"
2595 PRINT:PRINT:PRINT"                 PULSA UNA TECLA PARA VOLVER AL MENU PRINC
IPAL"
2597 LET Z$=INKEY$:IF Z$="" THEN GOTO 897
2599 GOTO 100
3000 PRINT:PRINT:PRINT
3010 PRINT "OTRA PROPIEDAD?"
3020 LET W$=INKEY$:IF W$="" THEN GOTO 3020
3030 IF W$="S" OR W$="s" THEN GOTO 3050
3040 IF A=1 THEN GOTO 200
3045 IF A=2 THEN GOTO 500
3050 IF A=1 THEN GOTO 1000
3060 IF A=2 THEN GOTO 2000
```

Modificaciones para otros equipos

El programa es válido para el IBM-PC, AMSTRAD y MSX.

COMMODORE

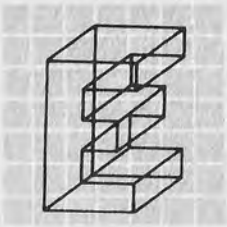
Cambiar los CLS por PRINT CHR$(147)

SPECTRUM

165 IF A=0 THEN GO TO 9999
180 IF A=1 THEN GO TO 200
182 IF A=2 THEN GO TO 500
185 IF A=3 THEN GO TO 700
188 IF A=4 THEN GO TO 900
1130 GOTO 1200 +(P*100)
2090 GOTO 2100 +(P*100)

PEQUEÑA HISTORIA DE LA INFORMATICA

Historia de las máquinas de calcular del siglo XVII



EN el siglo XVII aparecen los primeros intentos de simplificar las tareas de calcular mediante máquinas mecánicas. Antes, existían numerosos sistemas la mayoría muy ingeniosos para contar, pero ninguno era mecánico. Todos eran manuales, utilizando marcas, mediante tablas (logaritmos), reglas de cálculo, etcétera.

Ya en estas fechas aparece por primera vez la diferenciación entre los dos sistemas de cómputo, el analógico y el digital. Una regla de cálculo, por ejemplo, representa un sistema analógico de contar, ya que siempre se le pueden añadir más divisiones a la regla, o tomar otra más larga, de más precisión, etc., y el resultado no es un número concreto. Las máquinas mecánicas, por el contrario, son máquinas digitales, que utilizan cantidades exactas, concretas, obteniendo resultados también exactos.

Volviendo a las máquinas mecánicas, el problema a resolver es, en primer lugar, la cuenta de objetos, que no parece tener demasiadas complicaciones, si se van contando, por ejemplo, mediante un diente por cada objeto. El siguiente problema que aparece es el ARRASTRE, de una posición a la siguiente más significativa. Naturalmente la máquina debe hacer este paso automáticamente, sin que el operador se ocupe.

Para solventar el asunto, se pueden colocal, por ejemplo, varias ruedas engranadas directamente, pero con una reducción de diez a uno, de forma que por cada diez pasos o dientes de la primera rueda se produce un giro de un diente de la segunda. Los engranajes también pueden ser desmontables.

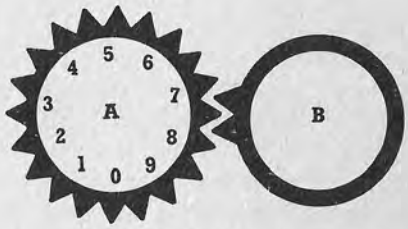


Fig. 1. Esquema básico de un contador mecánico que utiliza ruedas dentadas.

Pensemos, por ejemplo, en dos ruedas, A y B, una de ellas tiene veinte dientes, y la otra tiene solamente dos. Además, llevan escritos los dígitos de la forma en que se indica en la figura. Por cada vuelta completa de la rueda A, la rueda B gira un décimo de vuelta. Como los dígitos se pueden ver a través de la ventana, podremos leer una cantidad de dos dígitos, es decir, de 0 a 99. Si deseamos ampliar el sistema de cuenta, podemos hacerlo, engranando más ruedas, que sigan cumpliendo la norma anterior, es decir, que giren a su vez un décimo de vuelta por cada vuelta completa de la rueda anterior. Así, podemos continuar colocando ruedas, hasta donde queramos (al menos, desde el punto de vista teórico).

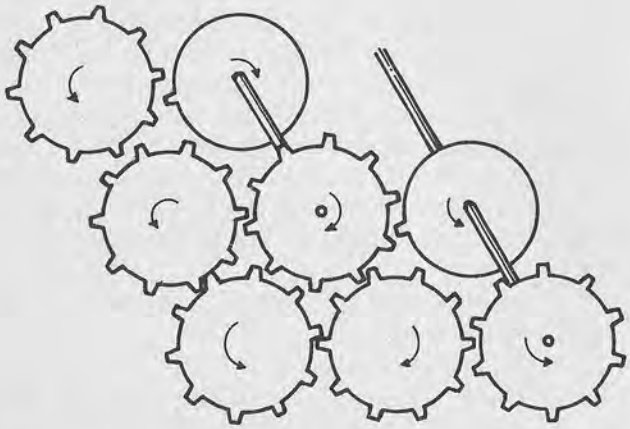
Hasta el momento, hemos fabricado una registradora, es decir, un aparato para llevar la cuenta. Ahora pasemos a sumar. Para ello, nuestros antecesores (o mejor, los antecesores de los franceses o alemanes) imaginaron dos contadores semejantes al que acabamos de describir. Al comenzar, los dos contadores es-

tán a cero, y el primero comienza a contar la cantidad. Inmediatamente, los contadores se vuelven a ajustar y giran juntos hasta anotar el segundo número. En el primer contador, se habrá anotado la suma de los dos números. Para llegar al resultado de forma práctica, se utiliza un trinquete. De este modo, el primer contador puede girar libremente, pero el segundo, en el momento en el que comienza a girar, arrastra SIEMPRE al primero. Las restas se efectúan girando la segunda rueda en sentido contrario.

Para que las operaciones resulten cómodas, es mucho mejor disponer de dos contadores, uno de ellos será el que registre los datos, y el otro anotará los resultados. Las sumas se pueden realizar mediante los engranajes con trinquete de que hablamos, de forma que los dos contadores giren solidarios en un determinado sentido, y se desconecten cuando giran en el sentido contrario, poniéndose a cero.

Resumiendo todo lo anterior, la máquina típica de Pascal estaba formada por varios mecanismos de ruedas, unas para registrar los datos, y otras para presentar los resultados. Cada rueda de registro de datos está unida a una rueda de resultados, y a su vez, lo está a las restantes ruedas que componen el mecanismo, siguiendo las normas anteriores sobre los arrastres.

Una vez esbozado el funcionamiento técnico de la máquina, pasemos a hablar de su autor. A la edad de diecisiete años, Blaise Pascal (1623-1662) concibió su primera máquina de calcular. Pascal, como todo científico era curioso, no aceptaba las cosas sin más, y gustaba de experimentar. Su padre era recaudador de Hacienda, y tenía que hacer penosísimos cálculos, y además, el Estado tampoco respondía totalmente de sus errores. Como el



Las ruedas dentadas tienen diez dientes o uno sólo, porque el sistema de numeración es decimal.

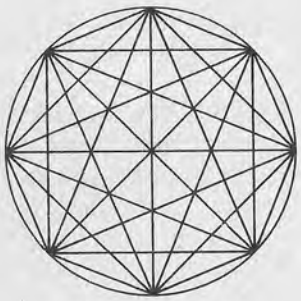
¿Sabía usted que...

Raimundo Lulio, padre de la Lógica Computacional Moderna, pensaba que para explorar el conocimiento, podía partirse de las siguientes consideraciones?: en cada rama del conocimiento existe un cierto número (no muy grande) de principios básicos que deben ser admitidos sin discusión, sin explicación de ningún tipo. Para llegar a explorar la Rama del Conocimiento seleccionada, un buen sistema es agotar todas las combinaciones posibles de esas categorías.

Para abordar el problema, Lulio pensó en construir tablas con todas las posibles combinaciones. Lulio ofrecía tres sistemas básicos: uno de ellos creando listas de dos conjuntos de categorías, en forma de columna, y uniéndolas rectas (tal y como mostramos en un dibujo). Otro sistema (que también mostramos en un dibujo) consiste en disponer el conjunto de categorías en un círculo, y trazar las rectas que unen dos categorías. Al hacer girar el círculo, se van obteniendo todas las permutaciones de dos términos.

El tercer sistema es el más típico de Lulio, y ha sido representado en multitud de grabados de la época. El método consiste en dibujar varios círculos concéntricos con los términos. Una vez hecho, se gira el círculo obteniéndose una magnífica tabla de combinaciones.

En la Edad Media, esos círculos se hacían de metal o de pergamino, y se pintaban de colores brillantes. Dada la incultura general, existente entre las gentes sencillas, el giro, el colorido y la forma de presentar el problema, proporcionaban una connotación esotérica a todo el trabajo, aumentando la fama de "extraño brujo" que tenía Lulio.



trabajo de su padre era tediosísimo, Pascal, siempre interesado por las Matemáticas, se aprestó a ayudarle "a su estilo", el estilo de un investigador muy bien dotado.

Sin haber estudiado documentación alguna que le sirviera de base (al contrario que Leibniz, del que hablaremos también), Pascal puso manos a la obra y diseñó una máquina de calcular. La máquina estaba bien concebida, pero otra cosa diferente era llevarla a la práctica. Eran muchas ruedas, y la transmisión del movimiento desde el comienzo hasta el último mecanismo constituía un problema complicado en la práctica. Pascal utilizó trinquetes, que iban subiendo hasta llegar al nueve en que caían a cero, avanzando la rueda siguiente una unidad. Ese momento era el más difícil, para el que existía una resistencia mayor.

En cuanto a la resta, Pascal pensó que el giro en el sentido contrario era muy complicado y podía crear problemas. Y diseñó la resta de forma que lo que se efectuaba era la suma de los números complementarios de los datos. De este modo, ambas operaciones se podían realizar con el mismo movimiento.

La multiplicación y división, ya no estaban tan ingeniosamente pensadas. En realidad, las multiplicaciones eran sumas reiterativas.

Pascal llegó a comercializar sus máquinas, lo que tiene mucha importancia, ya que diseñar dispositivos requiere ingenio, pero para llegar a dar vida a una máquina hace falta además de ingenio, tesón. La máquina de Pascal era suficientemente segura, como para repetirla, y venderla. (En 1960 todavía se podían encontrar máquinas de calcular basadas en el Pascalina.)

Sin embargo, sólo llegó a vender treinta y siete unidades.

Como antes dijimos, Pascal se interesó por la máquina hasta el punto de comercializarla. Para ello tuvo que localizar artesanos capaces de construirla a mano, y con la mayor fiabilidad. Los relojeros de la época eran ideales para este tipo de trabajos. Estaban acostumbrados a ingenios de relojería complicadísimos que daban la hora, el mes, la posición de los planetas, desfiles de muñecos, sonidos complicados, etc. Los buenos artesanos no escaseaban en aquella época, aunque los materiales no estaban a la altura de la habilidad de los que los domaban. Las máquinas resultaban caras, además, no hay que olvidar que el trabajo humano no tenía demasiado valor, de forma que aunque tuvieran que trabajar varios hombres para realizar un cálculo, la idea de que la máquina les sustituyera no tenía un interés especial, más bien su principal interés estribaba en "aliviar" el trabajo de algún contable suficientemente poderoso como para comprarla.

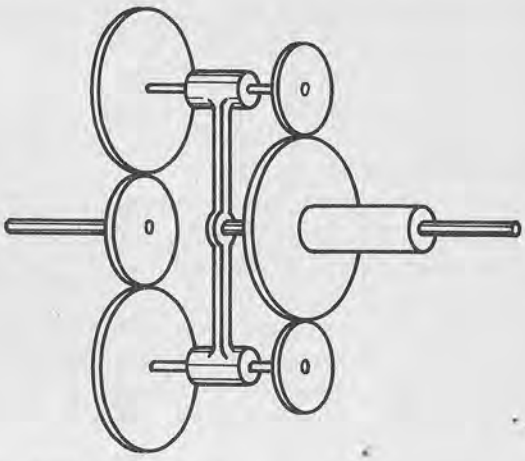


Fig. 3. Sumador diferencial.

El prototipo de la máquina de Pascal tardó siete años en construirse.

La máquina de Pascal supuso una pequeña revolución. La Pascalina era "capaz de realizar cálculos aritméticos sin errores, cosa difícil, incluso para personas que supieran mucha aritmética". (Estas líneas las escribió la hermana de Blaise Pascal, y constituyen una de las primeras reflexiones sobre lo que hoy se denomina "Inteligencia Artificial".)

La máquina de Pascal era ante todo innovadora. Pascal la desarrolló sin conocer los trabajos de Shickard (del que hablaremos a continuación) que a su vez se había basado en unos estudios de relojeros chinos anteriores al primer milenio antes de Cristo, pero ya hablaremos de ello más adelante. Además, su máquina realizaba algo nada usual en la época, hacía conversiones entre las distintas monedas en uso.

Es importante hacer notar que Pascal fue un investigador nato. Además de matemático de primera fila (cálculo de probabilidades, geometría descriptiva, etc.), fue un físico importantísimo. Estudió el comportamiento de los fluidos, la presión, y hoy en día sus experimentos y estudios siguen teniendo la misma vigencia. Entre sus aportaciones a las técnicas de vanguardia, está su prensa hidráulica, en la que se basan todos los sistemas modernos de

transmisión utilizados en miles de aparatos e instalaciones de nuestra vida cotidiana.

Entre sus invenciones y estudios sencillos, las cajas registradoras de hace unos años son máquinas casi idénticas a la máquina de calcular de Pascal.

Pascal construyó cincuenta máquinas, algunas de las cuales, verdaderas piezas de museo, se pueden ver en París, y en Clermont Ferrand su ciudad natal.

Otros inventores ya habían diseñado máquinas de calcular, que incluso multiplicaban y dividían. Pero no pasaron de ser prototipos interesantes. En 1623, en Alemania, William Shickardt diseñó una máquina muy lograda, llegando a construir el prototipo. Shickard era profesor de Matemáticas y Astronomía y hebreo en la Universidad de Tubinga. Sus estudios en este idioma le permitieron recoger ciertos trabajos realizados por relojeros chinos anteriores al primer milenio. Estos trabajos, pasaron a Oriente Medio, y de allí con los Cruzados a Europa. Sin embargo, su máquina se destruyó por un incendio, y nunca volvió a reconstruirse (realmente esto no es del todo cierto, ya que IBM rehizo la máquina partiendo de los escritos del autor. Uno de los ejemplares se puede ver en el Ayuntamiento de Tubinga, su ciudad natal. Shickard no tuvo demasiada suerte, y murió poco después de la peste con toda su familia, durante la Guerra de los Treinta Años. Se sabe muy poco de su existencia, salvo por algunos documentos salvados que permitieron reconstruir alguna de sus máquinas, y unas cartas enviadas al científico Johannes Kepler.

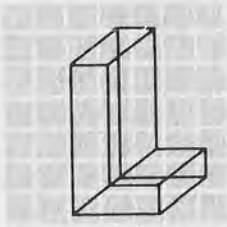
Su máquina permitía desconectar los engranajes, e introducir los números manualmente (permitiendo multiplicaciones rapidísimas para la época). Algunos de los diseños de máquina (construyó varios tipos) llevaba un cilindro con las Tablas de Logaritmos de Neper.

Algo más tarde, en 1663, apareció otra máquina en Inglaterra, diseñada por Samuel Moreland, que realizaba cálculos trigonométricos. Moreland construyó varias máquinas, algunas de ellas, muy semejantes a la de Pascal.

El diseño tan simple de estos "ingenios mecánicos" los ha hecho válidos hasta nuestros días. Los nuevos materiales, mucho más fiables, han contribuido notablemente a que fueran más rentables con el tiempo. En épocas recientes se han visto en las tiendas unos plumieres que llevaban incorporada una pequeña reproducción en plástico de máquina de calcular.

Los cuentakilómetros de los coches también llevan mecanismos muy semejantes.

Los ordenadores en el control del acelerador de partículas



A herramienta básica de trabajo de la física de las partículas es el acelerador. Se trata en principio de un acelerador de protones. Un acelerador consiste esencialmente en un conjunto de electroimanes y cavidades organizados alrededor de un anillo circular de 200 metros de diámetro. El rayo de protones circula atravesando los electroimanes por una conducción tal que las interacciones entre el rayo de protones y los gases residuales se reduce mucho. Existen, naturalmente, fuertes campos magnéticos, y pueden ser de dos tipos: dipolos, que doblan el rayo para mantenerlo en una órbita circular, y cuadripolos, que lo dirigen. El acelerador recibe unos 10 a 13 protones, para acelerarlos a 26 GeV en 2,4 segundos, que es lo que tardan en dar unas doscientas mil vueltas al acelerador. Como puede suponerse, el ajuste fino de todos los parámetros es delicadísimo. Los protones se extraen del acelerador mediante un tipo especial de electroimán y pueden pasar a utilizarse como rayos para otros trabajos, o bien pasar a otros aceleradores (algunos llegan a tener una circunferencia de varios kilómetros).

El acelerador no sólo puede acelerar protones, también iones pesados, electrones positivos y negativos, y otras partículas, para pasarlas a otros aceleradores más específicos.

Hace veinte años, durante el primer período de operación del acelerador del CERN (Centro Europeo que estudia la Física de las Partículas), una de las más importantes cosas que se mostraban a los visitantes era el cuarto de control de las máquinas. Esta habitación estaba totalmente llena con hileras de equipo electrónico, con matrices enteras de diales, muchísimos pilotos y pantallas de televisión. La tarea de los operadores consistía en llevar a cabo los ajustes manuales de este complicado equipo, para poner en marcha el acelerador, y mantenerlo en operación.

El cuarto de control del acelerador, construido hace aproximadamente una década, presentaba un aspecto totalmente diferente: consistía en una habitación amplia, ocupada por numerosas consolas, en la que estaban sentados numerosos operadores que controlaban el acelerador mediante preciosas presentaciones en color, pantallas táctiles, joysticks, pulsadores y teclados, todos ellos unidos a un sistema de ordenadores integrados. Todo ello como en las películas de ciencia ficción. Hablando seriamente, los cambios en estas habitaciones de control muestran el avance revolucionario de la Informática. Además, los precios han caído en picado, y son numerosas las tareas que antes se llevaban a cabo manualmente, y que ahora pueden efectuar los ordenadores. La capacidad de unirlos también ha ampliado sustancialmente el campo de su utilización. Todo ello, sin mencionar controles y tareas que antes eran inimaginables, mientras que ahora, con la ayuda del ordenador, se realizan sin especiales problemas. Citemos algunas de ellas.

1. Operación remota del equipo y adquisición de los datos.

2. Control del rayo, bien en bucle abierto, en el que el operador tiene cierto control basado en información que le presente el

sistema, o en bucle cerrado, en el que las condiciones requeridas están fijadas por el operador y el ordenador es el que se encarga de mantenerlas.

3. Cálculo de las condiciones de operación óptimas. Estos cálculos son complicados, porque manejan enormes cantidades de parámetros. Esta operación debe llevarse a cabo con rapidez y seguridad, porque de ello depende, entre otras cosas, la seguridad del experimento.

4. El registro de los parámetros que se han dado a la máquina para comprobar que ésta está funcionando satisfactoriamente.

5. Guía del operador en procedimientos automáticos.

6. Comprobaciones internas y control de acceso, en el cual la protección del equipo y el acceso del personal a zonas radiactivas está comprobado por el ordenador.

7. Transmisión de datos de las condiciones del rayo a experimentos, para su incorporación en sus sistemas de toma de datos.

En los antiguos sistemas de control del acelerador, la configuración típica del ordenador era sencillamente una con enorme potencia. Posteriormente, este enorme ordenador pasó a controlar a otro grupo de ordenadores pequeños, cada uno de los cuales controlaba una pequeña parte del acelerador. Con el tiempo, los pequeños ordenadores han ido complicándose para alcanzar mejores cotas de fiabilidad. También están funcionando sistemas con inteligencia artificial, y este punto es muy importante: no basta con que el ordenador detecte el error que se está produciendo, es necesario que tome las medidas oportunas, como, por ejemplo, ejecutar un programa de diagnóstico, para averiguar las razones de la dificultad detectada y permitir que operadores que se encuentran a varios kilómetros de la central tomen las medidas oportunas. Se han utilizado mucho la familia de ordenadores construidos sobre Motorola 68.000, junto con VME bus interconexiones.

La red de ordenadores más adecuada para el acelerador tiene dos niveles. Unos ordenadores se utilizan para el proceso, y controlan el equipo al nivel más inferior. No es útil que éstos sean miniordenadores, sino un conjunto de microprocesadores, configurados sencillamente para que cumplan los requerimientos, y basados en lo que se llama arquitectura función a función. Cada microordenador tiene una tarea específica, y se comunica con los demás, pasando mensajes y también con el equipo a través de un canal de gran capacidad. Los controladores del proceso están unidos unos a otros al nivel más alto, a través de un sistema de comunicaciones basado en un multiplexor de fibra óptica, que también es utilizado por el sistema de control de tiempos y para teléfonos digitales y conexiones vía vídeo.

Cada una de las partes del equipo que componen el acelerador, los imanes, las fuentes de energía, etc., llevan asociada la correspondiente información sobre los parámetros, las entradas y el sistema de operación. El resultado es un enorme cúmulo de datos, que deben ser procesados del mejor modo posible: es una decisión importante. La técnica seleccionada es distribuir los datos: parte en una gran base de datos, ubicada en el sistema central, y parte sobre todo el sistema de microprocesadores y ordenadores del sistema. La decisión entre los dos niveles se tomará considerando si la información corresponde a un elemento del equipo o es común a todos los que están unidos al ordenador de proceso central. Lógicamente, cada pieza del equipo está asociada con uno de estos módulos, que a su vez está dividido en dos trozos, tal y como hemos descrito anteriormente. También se guardan copias de esta base de datos para propósitos de archivo o recarga, pero debe haber una base de datos central, involucrada en el funcionamiento global de la máquina de forma que proporcione los parámetros estándar correspondientes a las condiciones generales de operación. Esta base de datos central también permitirá el ajuste directo de los parámetros del rayo, no únicamente los de la máquina. Por ejemplo, un operador puede querer establecer el ángulo de un rayo, y a continuación permitir a la base de datos que proporcione los parámetros correspondientes a ese ángulo, en lugar de tener que controlar todos los dispositivos necesarios para obtener el ángulo necesario.

Los sistemas principales de control del acelerador están programados en lenguajes de alto nivel; el acelerador de protones en un lenguaje derivado del PASCAL llamado P+, y también se utiliza otro lenguaje desarrollado a partir del FOCAL, y con ciertas características del BASIC, y SNOBOL 4. Se llama NODAL (y ha sido desarrollado por NORD-10). Este lenguaje se utiliza también en otro tipo de laboratorios. Dispone de la facilidad de poder ejecutar partes de un programa en otros orde-

nadores, para esperar los resultados. La posibilidad de utilizar lenguajes de alto nivel es importantísima, ya que han podido entrar a formar parte del equipo de programación ingenieros y físicos, que conocen el diseño y construcción de la máquina, y, en suma, son profesionales especializados, en lugar de utilizar simplemente programadores, que difícilmente llegarían a comprender con tanta profundidad las necesidades y posibilidades de la máquina. Estos lenguajes han sido diseñados por CERN (Laboratorio Europeo para la Física de las Partículas), y en cuanto a los sistemas operativos, los fabricantes de los ordenadores utilizados, son los que suelen realizar las mejoras. Existen diferencias entre los EE.UU. y Europa, donde los sistemas de control están basados en ordenadores VAX que utilizan el sistema operativo VMS, con programación de alto nivel, llevada a cabo en FORTRAN 77. De hecho, debemos decir que existen casi tantos sistemas de control del acelerador como aceleradores mismos.

Se está en este momento en la fase de implantación de un nuevo lenguaje, el MODULA-2, para escribir el intérprete de NODAL y mucho del software básico. Este lenguaje proporciona un sistema de software portátil, es decir, posibilita el cambio del hardware (ordenador propiamente dicho) sin necesidad de revisar el software, situación muy común, ya que la vida de un acelerador es varias veces más larga que las de un ordenador dado.

Los recursos necesarios para construir y hacer funcionar un laboratorio de física nuclear son considerables. En sus primeros comienzos era frecuente que fueran decisiones a nivel de Estado, o también, como ocurre en ciertos países a nivel de varias Universidades que se unen para disponer de un laboratorio conjunto. Sin embargo, con el enorme aumento de las posibilidades, han aumentado paralelamente los gastos, que son ya tan enormes que requieren colaboración internacional. De este modo, se considera que es más rentable construir menos aceleradores, pero más grandes. Con esta idea, podemos citar en Europa los de CERN y DESY; en EE. UU. Brookhaven Fermilab, Cornell, y en la URSS, Dubna y en Japón KEK. Existen organizaciones internacionales que deciden y recomiendan las posibles ubicaciones y tipos de los futuros aceleradores, para obtener un mayor grado de cooperación, procurando que no se repitan trabajos que se han realizado en otra central y evitar la duplicidad. El estilo de trabajo varía de unos a otros, pero los objetivos son idénticos.

En Europa Occidental, el CERN es un laboratorio internacional, creado por varios Estados para satisfacer sus necesidades. Tiene una dotación de científicos fijos, y otros itinerantes, que van variando año tras año.

Los experimentos propiamente dichos son llevados a cabo por colaboraciones entre varios cientos de científicos.

Los gastos del laboratorio también son enormes. Y aunque los trabajos están basados en la Física pura, el empuje de la industria hace que siempre estén ligados en cierto modo a ella.

Para cada experimento adjudicado al equipo de algún científico de prestigio, existe un comité que vigila que el dinero no se derroche, y se adjudiquen trabajos interesantes a personas que los pueden sacar adelante.

ALGUNOS TIPOS DE ACELERADORES Y TERMINOS RELACIONADOS CON ELLOS

Acelerador. Dispositivo que emplea campos magnéticos y eléctricos de alta frecuencia para producir aceleraciones en las partículas en sincronismo cientos de miles de veces. Tanto la órbita como la intensidad del haz se mantienen constantes. Los primeros aceleradores de iones consistían simplemente en tubos de descarga en los que se hacía el vacío y a los que se aplicaban tensiones continuas.

Acelerador lineal. Dispositivo que acelera partículas a altas energías mediante un campo eléctrico oscilante. Se utiliza principalmente para acelerar iones pesados a ciertas energías (3MeV o similares).
El acelerador elemental consistía en un conjunto de tubos metálicos conectados a una fuente de potencia de alta frecuencia, y colocados dentro de una cámara de vacío. Al reproducirse una descarga en un extremo del acelerador las partículas se aceleraban al atravesar el espacio entre los dos electrodos. En los aceleradores lineales también se cumple el principio de estabilidad en fase. En general, resultan más prácticos los sincrotrones, que disponen del electroimán en forma de anillo.

Betatrón. Dispositivo que data de 1939. Consiste esencialmente en una cámara de vacío en la que se encuentra una nube de electrones libres, que hace el papel de devanado secundario de un transformador. Los electrones se mueven en órbitas de radio constante, y van ganando energía por inducción (debido al flujo del campo magnético variable). Los electrones hacen su trayectoria cientos de miles de veces ganando unos cientos de voltios por vuelta.

Ciclotrón. Dispositivo construido por Lawrence y Livingston en la Universidad de California en 1932. Aceleraba partículas cargadas hasta 1 meV (hoy en día se obtienen energías mucho más altas (20 meV e incluso más). El sistema introducía el concepto de aceleración múltiple, obtenida al disponer un campo eléctrico oscilante. De este modo, no era necesario utilizar potenciales tan altos. Este concepto se aplicó a partículas con órbitas circulares producidas en un campo magnético.

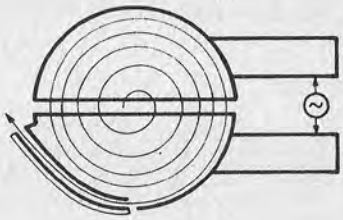


Fig. 1. Esquema de un ciclotón.

Esencialmente, un ciclotrón consiste en dos electrodos huecos en forma de D montados en un tubo vacío. Están conectados a una fuente de tensión de alta frecuencia, y se colocan perpendicularmente a un campo magnético fuerte. La fuente de iones se encuentra en el centro entre los dos electrodos. La partícula sólo sufre el efecto del campo magnético, excepto en el momento en el que cruza la separación existente entre las dos D. El tiempo que tarda el ion en su trayectoria por el semicírculo debe ser igual al que tarda el campo eléctrico en cambiar de signo. De este modo, la partícula va incrementando su energía en 2 e V cos alfa. La partícula efectúa muchísimas tra-

yectorias por el acelerador, aumentando su energía en cada vuelta. Los ciclotrones de nuestros días utilizan fuentes de alta frecuencia y tensiones muy altas, con el fin de reducir el número de revoluciones.

Estabilidad de fase. El radio de la órbita de una partícula es función de su masa y del campo magnético. Su frecuencia angular es igual a la frecuencia del oscilador para ese radio.
Las partículas cuya energía inicial es ligeramente diferente de la de equilibrio describen órbitas de radios mayores o menores que los de equilibrio. Sus frecuencias también serán diferentes de la del campo eléctrico. Sin embargo, si se producen cambios no muy importantes, las partículas tienden a perder o ganar la energía suficiente y restablecen las condiciones de equilibrio. Si se va aumentando continuamente el campo magnético, o si se disminuye la frecuencia, las partículas irán ganando energía constantemente a una velocidad determinada, por la variación de la frecuencia o del campo magnético. (En esta característica se basa otro dispositivo, el sincrotrón.)

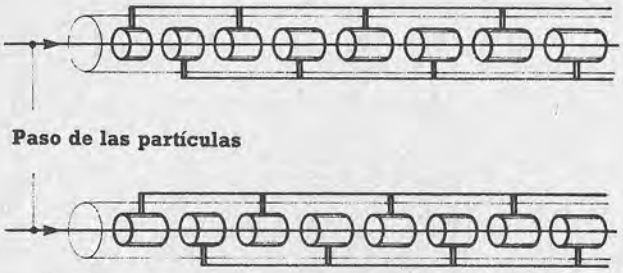


Fig. 2. Esquema de un acelerador lineal.

Fuentes de tensión. En un principio se utilizaron tensiones continuas. Las fuentes utilizadas consistían simplemente en un conjunto de transformador rectificador. Más adelante, se utilizaron generadores de alta tensión.

Oscilaciones. Pequeñas variaciones entre la órbita de equilibrio de las partículas. Pueden ser radiales y verticales. El campo magnético debe calcularse para que éstas sean mínimas.

Sincrociclotrón. En los ciclotrones, la energía se veía en cierto modo limitada por el aumento de la masa (teoría de la relatividad). La frecuencia del ion disminuía, y éste

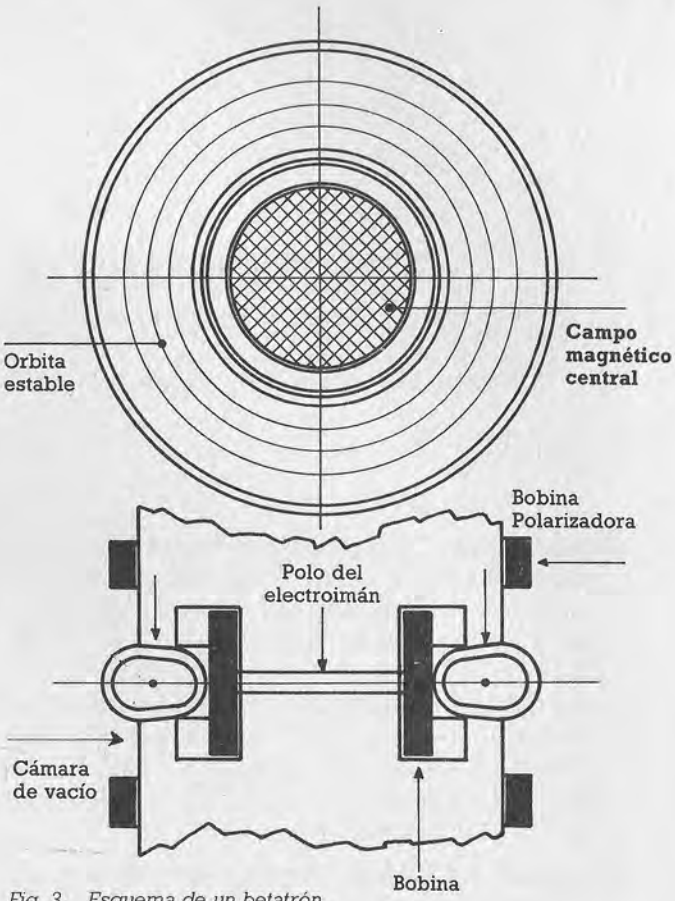


Fig. 3. Esquema de un betatrón.

dejaba de resonar con el campo eléctrico aplicado.
En un ciclotrón, el radio de la órbita estable es el obtenido cuando la frecuencia angular del ion es iugal a la frecuencia del oscilador. De este modo, aumentando enormemente la frecuencia, el radio puede disminuir hasta alcanzar un valor límite dado por la fórmula.

$$w_o = \frac{e}{m_o} - B_c$$

siendo m_o la masa en reposo del ion, y B_c el campo aplicado a la fuente de iones.
Si se hace disminuir la frecuencia del oscilador hasta w_0 - dw, el radio de la órbita estable aumenta. Cuanto más rápidamente aumente la frecuencia, más energía obtendrá la partícula por cada ciclo, y por eso necesitará menos ciclos para alcanzar el radio deseado.
La modulación de la frecuencia del oscilador se realiza con un condensador que va girando a velocidad variable. La potencia necesaria para el oscilador es mucho menor que en el ciclotrón, ya que las partículas giran muchas más revoluciones. El sin-

crociclotrón es capaz de acelerar protones, con energías muy altas, capaces a su vez de crear mesones por colisiones nucleares. No son válidos, sin embargo, para acelerar electrones, ya que éstos tienen una masa muy pequeña. Los electrones se aceleran mediante el sincrotrón.

Sincrotrón. Método de acelerar electrones que utiliza un campo magnético anular producido por un electroimán (en forma de anillo). Acelera los electrones en una órbita de radio constante. Una de las ventajas sobre el betatrón es el ahorro de peso que supone la utilización del electroimán. El principio utilizado en este dispositivo es el mismo que el del sincrociclotrón: la frecuencia de las partículas cargadas debe ser igual a la del oscilador. Para ajustarlo, se varía la frecuencia de éste. En la práctica, como los electrones alcanzan velocidades enormes, es más útil dejar fija la frecuencia del oscilador y variar el campo magnético. Un sistema que se ha utilizado es acelerar las partículas hasta ciertas velocidades, para más tarde pasarlas al ciclotrón, que las acelera hasta valores muy superiores. El límite de energía en el sincrotrón es, pues, algo más alto que en el betatrón.

Fichero, mantenimiento de un. Acciones tomadas para mantener al día el fichero, introduciendo, borrando o cambiando los datos que contiene.

Fichero secuencial. Fichero cuyos registros se han ordenado siguiendo un criterio secuencial (uno detrás del otro). Para acceder a un determinado registro, debe «pasarse» por los anteriores.

Fichero, separador de. Elemento que sirve para separar los distintos elementos de información contenidos en el fichero.

FIFO (First In First Out). (Primero en entrar, primero en salir). Sistema de funcionamiento en una cola de espera, según el cual es el primero que llega el primero que se ejecuta.

Figurativa, constante. En un lenguaje de programación, cadena de caracteres fija, previamente asignada, y con nombre fijo.

File. (Ver **Fichero.**)

File gap. Zona de un medio que se utiliza para indicar el final de un fichero, y probablemente el comienzo de otro. También puede utilizarse para indicar el comienzo o el fin de algún grupo de datos.

Filtro. Dispositivo o programa que sirve para separar datos, señales o cualquier otro elemento o elementos, según un determinado criterio. También puede referirse a una máscara.

Firmware. Término inglés utilizado para designar el software incluido en una memoria ROM, y, por tanto, incorporado a la máquina.

Flag. Término inglés utilizado para designar a un indicador, es decir, para indicar si una determinada situación es verdadera o falsa. En la mayoría de los casos, podría ocupar un bit de memoria, pero ocupa un byte como toda las variables lógicas.

Flip-flop. Circuito basculante, es decir, capaz de tomar en un momento dado uno de los dos estados estables.

Flujo. Secuencia de acontecimientos. También se utiliza el término en la denominación de un diagrama de flujo, diagrama en el cual se indican de forma esquemática las operaciones desarrolladas en un programa. Puede referirse, por otro lado, a una secuencia ordenada de registros.

Flujo bidireccional. En un diagrama de flujo, aquél que se puede extender en las dos direcciones de una línea de flujo.

Flujo, línea de. Línea que representa una transferencia o control de los datos, en un diagrama de flujo.

Font. (Véase **Fuente.**)

Foreground. En multiprogramación, suele referirse a una de las divisiones efectuadas en la memoria central. La primera zona se denomina foreground, o memoria de «pri-

mer plano», frente a la segunda, background, «memoria de segundo plano». Los programas que se encuentran en la primera zona se ejecutan en tiempo real (frente a los segundos, que lo hacen en proceso batch).

Foreground processing. Ejecución automática de programas, según criterios establecidos con anterioridad desde la zona de foreground. Suele utilizarse para designar a los programas que se ejecutan en tiempo real. En contraposición con procesos batch.

Form feed. Carácter que controla el movimiento de la posición de impresión, pasando a la página siguiente. Se utiliza muy frecuentemente su abreviación, FF.

Formato de un fichero. Disposición que toman los datos en un fichero. En ella debe considerarse la estructura, la secuencia y el tamaño de sus componentes.

Format effector. Carácter de control utilizado por un dispositivo de presentación o de impresión, y que controla la presentación o la posición de la información. Con frecuencia se utiliza su abreviación, FE. Sinónimo de carácter de presentación.

Formatear. Operación realizada en lo discos rígidos o flexibles la primera vez que se van a utilizar. Consiste en grabar en ellos todo aquello que posteriormente servirá para su empleo (directorió, comienzos de los sectores, etc.) También se pueden formatear los discos ya utilizados, pero en ese caso se perderá toda la información almacenada en ellos. Este tipo de información grabada depende del sistema operativo de cada ordenador, por lo que un disco formateado con un determinado ordenador puede no servir para otro.

Frame. Término inglés utilizado para referirse a un área de una longitud igual a una posición de grabación y que se extiende a lo ancho de una cinta magnética, perpendicular a su movimiento. Se pueden incluir en un único «frame», en ocasiones varias posiciones de grabación.

Fuente. Conjunto de caracteres (disponibles en una impresora o dispositivo de presentación gráfica) de un determinado tamaño, estilo, etc. Por ejemplo, letra inglesa o itálica, etcétera.

Full duplex. (Véase **Dúplex.**)

Función. Vínculo entre una o más variables independientes y una o más variables dependientes. De este modo, a cada valor le corresponde, por la función, otro u otros valores. En programación, en el lenguaje que se utilice, están establecidas generalmente algunas funciones, aunque el programador puede definir las que le parezca oportuno (en la mayoría de los lenguajes).

Función, tabla de. Cuando dos o más conjuntos de datos están organizados de forma que una entrada en uno de ellos selecciona otra o más entradas en los restantes conjuntos. Tabulación de los valores de una función para un conjunto de valores de la variable. También se puede referir a un dispositivo de hardware o a una subrutina que puede decodificar entradas múltiples en una salida única o codificar una entrada única en salidas múltiples.

Funcional, diagrama. Diagrama que representa las relacionales funciones entre las distintas partes de un determinado sistema.

Funcional, diseño. Especificación de las relaciones de trabajo entre las distintas partes del sistema, considerando sus acciones características.